Progress® DataDirect®
Hybrid Data Pipeline
User's Guide

Release 4.6
Copyright

© 2019 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

Corticon, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Deliver More Than Expected, Icenium, Kendo UI, Kinvey, NativeScript, OpenEdge, Powered by Progress, Progress, Progress Software Developers Network, Rollbase, SequeLink, Sitefinity (and Design), Sitefinity, SpeedScript, Stylus Studio, TeamPulse, Telerik, Telerik (and Design), Test Studio, and WebSpeed are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries. Analytics360, AppServer, BusinessEdge, DataDirect Autonomous REST Connector, DataDirect Spy, SupportLink, DevCraft, Fiddler, JustAssembly, JustDecompile, JustMock, NativeChat, NativeScript Sidekick, OpenAccess, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

Updated: 2019/07/25
# Table of Contents

**Welcome to DataDirect Hybrid Data Pipeline**............................................17  
- Product requirements............................................................................................................................18
- Deployment guidelines..........................................................................................................................21
- Deployment scenarios..........................................................................................................................22
- Standalone deployment......................................................................................................................23
- Load balancer deployment..................................................................................................................36
- Exposing on-premises data sources to cloud-based applications.............................................51
- Connecting an application in the cloud to on-premises data sources........................................52
- External JRE support and integration........................................................................................52

**Administering Hybrid Data Pipeline**...........................................................57  
- Initial login with default user accounts...........................................................................................58
- Permissions and default roles............................................................................................................59
- Logging in to the Web UI...................................................................................................................61
- Using Hybrid Data Pipeline APIs........................................................................................................62
- Using the Web UI..................................................................................................................................63
- Manage Tenants view......................................................................................................................64
- Manage Users view..............................................................................................................................65
- Manage Roles view..............................................................................................................................67
- Data Sources view..............................................................................................................................69
- SQL Editor view...................................................................................................................................74
- Manage Limits view.............................................................................................................................76
- Manage External Authentication view..........................................................................................78
- System Configurations view.............................................................................................................79
- Product information.............................................................................................................................81
- User profile........................................................................................................................................81
- Changing your password in the Web UI.........................................................................................81
- Tenant architectures..........................................................................................................................82
- Single-tenant environment..............................................................................................................83
- Multitenant environment..................................................................................................................90
- User provisioning................................................................................................................................107
- Provisioning users with the Web UI..............................................................................................109
- Provisioning users with Hybrid Data Pipeline APIs..................................................................114
- Managing permissions with Hybrid Data Pipeline APIs..............................................................133
- Authentication.....................................................................................................................................143
- Integrating external authentication with a Java plugin.................................................................144
- Integrating an LDAP authentication service................................................................................153
- Advanced functionality for authentication services.................................................................157
## Contents

Password policy .................................................................................................................................. 159
Enabling and disabling the password policy ....................................................................................... 160
Configuring change password behavior .............................................................................................. 161
Implementing an account lockout policy............................................................................................. 161
Transactions ........................................................................................................................................ 163
Implementing IP address whitelists..................................................................................................... 164
   Configuring IP address whitelist ...................................................................................................... 165
   Enabling and disabling the IP address whitelist feature ........................................................... 168
Configuring row limit throttling .......................................................................................................... 169
Configuring throttling for OData queries ............................................................................................ 170
Configuring CORS behavior ............................................................................................................... 172
FIPS (Federal Information Processing Standard) ............................................................................... 173
   Enabling and disabling FIPS ........................................................................................................ 174
   Before enabling FIPS ................................................................................................................ 176
Data source logging............................................................................................................................ 176
   Setting data source logging levels ................................................................................................. 177
   The data source activity log ......................................................................................................... 181
Using third party JDBC drivers with Hybrid Data Pipeline ................................................................ 182
   Verifying a third party JDBC driver for Hybrid Data Pipeline compatibility .................................. 183
   Integrating the third party JDBC driver into Hybrid Data Pipeline ............................................. 185
Configuring Hybrid Data Pipeline to authorize client applications using OAuth 2.0 ....................... 185
   Establishing an OData connection using OAuth 2.0 ...................................................................... 185
   Register client application ............................................................................................................ 186
   OAuth 2.0 tokens ......................................................................................................................... 186
   OAuth grant flows ....................................................................................................................... 187
   OAuth 2.0 endpoints .................................................................................................................... 189
Integrating Hybrid Data Pipeline with a Google OAuth 2.0 authorization flow to access Google Analytics .......................................................................................................................... 190
   Registering Hybrid Data Pipeline as a client application with the Google Analytics API .......... 191
   Using the OAuth applications API to create an OAuth application object .................................... 192
   Using the OAuth profiles API to create an OAuth profile ............................................................. 194
   Using the Data Sources API to create a Google Analytics data source ....................................... 196
Troubleshooting .................................................................................................................................. 196
   System logs .................................................................................................................................. 197
   MySQL Community Edition troubleshooting ............................................................................... 199
   Out of memory errors .................................................................................................................. 199
   IP address troubleshooting ......................................................................................................... 199
   Extracting schema files for non-relational data sources ............................................................ 199
   Contacting Technical Support ...................................................................................................... 200

### Using Hybrid Data Pipeline

- Logging in to the Web UI .................................................................................................................. 203
- Using Hybrid Data Pipeline APIs .................................................................................................. 204
- Using the Web UI .......................................................................................................................... 205
Creating data sources with the Web UI........................................................................... 224
How to create a data source in the Web UI................................................................. 225
Supported data stores............................................................................................... 226
Amazon Redshift parameters...................................................................................... 227
Apache Hadoop Hive parameters............................................................................... 240
Autonomous REST Connector parameters.............................................................. 257
DB2 parameters....................................................................................................... 272
JDBC parameters for third party drivers................................................................. 292
Google Analytics parameters................................................................................... 298
Greenplum parameters............................................................................................ 318
Informix parameters.................................................................................................. 334
Microsoft Dynamics CRM parameters...................................................................... 345
Microsoft SQL Server parameters............................................................................ 358
MySQL Community Edition parameters.................................................................. 384
MySQL Enterprise parameters................................................................................ 390
Oracle parameters................................................................................................... 405
Oracle Marketing Cloud (Eloqua) parameters......................................................... 435
Oracle Sales Cloud parameters................................................................................ 449
Oracle Service Cloud parameters............................................................................ 462
PostgreSQL parameters.......................................................................................... 476
Progress OpenEdge parameters............................................................................... 492
Progress Rollbase parameters................................................................................ 506
Salesforce (and Related Data Store) connection parameters................................. 519
SugarCRM parameters............................................................................................ 592
Sybase parameters.................................................................................................. 606
Editing, testing, and deleting data sources with the Web UI....................................... 621
Editing connection parameters.................................................................................. 621
Testing a data source................................................................................................. 621
Deleting a data source............................................................................................... 622
Configuring data sources for OData connectivity and working with data source groups. 622
Configuring data sources for OData Version 2 connectivity...................................... 623
Configuring data sources for OData Version 4 connectivity...................................... 627
Creating a data source group..................................................................................... 635
Editing a data source group....................................................................................... 636
Contents

Deleting a data source group..........................................................637
Creating and using REST data sources.................................................637
Creating REST data sources with the API.............................................638
Creating an input REST file ...........................................................641

Configuring Hybrid Data Pipeline Driver for ODBC.........................649

Getting started with the ODBC Driver.................................................650
Supported features .............................................................................650
  Data encryption..............................................................................650
  Unicode.........................................................................................650
  Safe thread handling.......................................................................651
  Number of connections and statements supported..........................651
  Parameter metadata........................................................................651
  SQL support..................................................................................651
Configuring an ODBC data source on UNIX and Linux systems............651
  Setting environment variables manually.........................................653
  Configuring a data source in the system information file...................655
  DSN-less connections......................................................................657
  Example application for UNIX and Linux.........................................658
Configuring and testing an ODBC data source on Windows....................658
  Example application for Windows..................................................659
Connecting applications to the connectivity service............................659
  Connection strings ........................................................................659
  File data sources ..........................................................................660
  Connection Properties ....................................................................661
  Connecting Through a Proxy Server.................................................676
Connection properties reference........................................................676
  ODBC Connection Properties........................................................677
  Application Using Threads ..............................................................678
  Client Time Zone .........................................................................679
  Data Source Name .........................................................................680
  Data Source Password ....................................................................680
  Data Source User...........................................................................681
  Default Buffer Size for Long/LOB Columns (in Kb).............................682
  Description .................................................................................682
  Enable SSL..................................................................................683
  Enable WChar Support ..................................................................683
  Host Name In Certificate ................................................................684
  Hybrid Data Pipeline Source ..........................................................685
  IANAAppCodePage ......................................................................686
  Login Timeout ..............................................................................686
  Logon Domain ..............................................................................687
  Max Varchar Size ..........................................................................688
  Min Long Varchar Size ...................................................................688
Configuring Hybrid Data Pipeline for JDBC

Password ................................................................. 689
Port Number .............................................................. 690
Proxy Host ................................................................. 690
Proxy Password ......................................................... 691
Proxy Port ................................................................. 691
Proxy User ................................................................. 692
Query Timeout ........................................................... 692
Report Codepage Conversion Errors .......................... 693
Service ........................................................................ 693
Transaction Mode ...................................................... 694
Trust Store ................................................................. 695
Trust Store Password .................................................. 695
User Name .................................................................. 696
Validate Server Certificate ......................................... 697
Varchar Threshold ....................................................... 697
WSRetryCount ........................................................... 698
WSTimeout .................................................................. 699
Application considerations ........................................ 700
Verifying the driver version number ......................... 700
Retrieving data type information ............................... 702
Supported ODBC API functions ............................... 703
Scalar functions ......................................................... 706
Troubleshooting ......................................................... 708
Determining where an issue originates .................... 708
Log files created during installation or upgrade .......... 709
Error message syntax ................................................. 709
Test loading tools for UNIX and Linux .................... 711
ODBC Trace ............................................................... 712
Creating a trace log .................................................... 714
Other tools ............................................................... 715
ODBC Test ................................................................. 715
Using the Driver with Microsoft Access .................... 715
Internationalization, localization, and Unicode ......... 715
Internationalization and Localization ....................... 715
Unicode Character Encoding ..................................... 717
Unicode ODBC Drivers .............................................. 719
Driver Manager and Unicode Encoding on UNIX/Linux 721
References ................................................................ 722
Code page values ...................................................... 722
WorkAround options ................................................ 726
WorkArousnds and WorkArounds2 options ............... 726
Using the WorkAround options ............................... 728

Configuring Hybrid Data Pipeline for JDBC
Querying with OData Version 2

Getting started with OData Version 2
Contents

Configuring data sources for OData Version 2 connectivity ..................................................... 828
Testing data source configurations (OData Version 2) ............................................................. 833
Requesting service metadata and the service document ......................................................... 834
Supported functionality for OData Version 2 ........................................................................... 835
Supported OData operations and data types ........................................................................... 835
Understanding and configuring a schema map for OData Version 2 ....................................... 838
  JSON schema map syntax ....................................................................................................... 839
  Schema map examples ........................................................................................................ 841
Structure of requests for OData Version 2 ............................................................................. 842
  Headers ................................................................................................................................. 843
  Service URI and resource path ............................................................................................ 845
  Response formatting ........................................................................................................ 846
Formulating queries with OData Version 2 ............................................................................. 846
  Query options and optimizing response times ..................................................................... 847
Searching text-based columns with OData Version 2 ............................................................ 849
Creating, editing, and deleting records with OData Version 2 .............................................. 851
Navigating relationships with OData Version 2 ..................................................................... 855
Method Reference for OData Version 2 .................................................................................. 857
  HTTP GET ........................................................................................................................... 858
  HTTP DELETE or POST and DELETE ................................................................................ 859
  HTTP POST and MERGE (update) .................................................................................. 859
  HTTP POST (create) ......................................................................................................... 860

Querying with OData Version 4 ......................................................................................... 863
  Getting started with OData Version 4 .................................................................................. 863
    Configuring data sources for OData Version 4 connectivity ............................................. 864
    Testing data source configurations (OData Version 4) ................................................... 872
    Requesting service metadata and the service document ................................................. 875
  Supported functionality for OData Version 4 ..................................................................... 876
    Supported OData operations and data types ................................................................. 877
    Stored functions support ............................................................................................... 880
    OData model warnings ................................................................................................. 880
  Understanding and configuring a schema map for OData Version 4 .................................... 881
    JSON schema map syntax ............................................................................................ 882
    Schema map examples ................................................................................................. 886
  Structure requests for OData Version 4 ............................................................................. 888
    Headers ............................................................................................................................. 888
    Service URI and resource path in Hybrid Data Pipeline ............................................. 891
    Response formatting for OData Version 4 ..................................................................... 891
  Formulating queries with OData Version 4 ....................................................................... 892
    Query options and optimizing response times ............................................................. 892
    Searching text-based columns .................................................................................... 895
    Fetching records and collections .................................................................................. 896
Querying data stores with SQL ................................................................. 909

Querying data stores with SQL ................................................................. 909
Supported data types .............................................................................. 910
Entity Data Model (EDM) types for OData Version 4 ......................... 911
Entity Data Model (EDM) types for OData Version 2 ......................... 912
Amazon Redshift data types ................................................................. 913
Apache Hive data types ........................................................................ 913
Autonomous REST Connector data types ........................................... 914
DB2 data types .................................................................................... 916
Greenplum data types ........................................................................... 917
Informix data types .............................................................................. 918
Microsoft Dynamics CRM Online data types .................................... 920
Microsoft SQL Server data types ......................................................... 922
MySQL data types ............................................................................... 924
Oracle data types ................................................................................ 925
Oracle Marketing Cloud (Eloqua) data types .................................... 927
Oracle Sales Cloud data types ............................................................. 927
Oracle Service Cloud data types ......................................................... 928
PostgreSQL data types ........................................................................ 929
Progress OpenEdge data types .......................................................... 930
Progress Rollbase data types ............................................................. 931
Salesforce-type data types ................................................................. 933
SugarCRM data types ........................................................................... 935
Sybase data types ................................................................................. 938
Supported scalar functions ................................................................. 940
Scalar Function Support for Amazon Redshift ................................... 940
Scalar Function Support for Apache Hive .......................................... 941
Scalar Function Support for Autonomous REST Connector ............. 943
Scalar Function Support for DB2 ......................................................... 944
Scalar Function Support for Google Analytics ................................... 946
Scalar Function Support for Greenplum ............................................. 947
Scalar Function Support for Informix ................................................ 948
Scalar Function Support for Microsoft Dynamics ............................. 949
Scalar Function Support for Microsoft SQL Server .......................... 951
Scalar Function Support for MySQL .................................................. 952
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalar Function Support for Oracle</td>
<td>953</td>
</tr>
<tr>
<td>Scalar Function Support for Oracle Marketing Cloud (Eloqua)</td>
<td>954</td>
</tr>
<tr>
<td>Scalar Function Support for Oracle Sales Cloud</td>
<td>955</td>
</tr>
<tr>
<td>Scalar Function Support for Oracle Service Cloud</td>
<td>957</td>
</tr>
<tr>
<td>Scalar Function Support for PostgreSQL</td>
<td>958</td>
</tr>
<tr>
<td>Scalar Function Support for Progress OpenEdge</td>
<td>959</td>
</tr>
<tr>
<td>Scalar Function Support for Progress Rollbase</td>
<td>960</td>
</tr>
<tr>
<td>Scalar Function Support for Salesforce-based data stores</td>
<td>962</td>
</tr>
<tr>
<td>Scalar Function Support for SugarCRM</td>
<td>963</td>
</tr>
<tr>
<td>Scalar Function Support for Sybase</td>
<td>964</td>
</tr>
<tr>
<td>Using Salesforce reports</td>
<td>966</td>
</tr>
<tr>
<td>Salesforce data store reports</td>
<td>966</td>
</tr>
<tr>
<td>Supported SQL and Extensions</td>
<td>967</td>
</tr>
<tr>
<td>Alter Session (EXT)</td>
<td>970</td>
</tr>
<tr>
<td>Alter Table for Salesforce</td>
<td>971</td>
</tr>
<tr>
<td>Create Table for Salesforce</td>
<td>974</td>
</tr>
<tr>
<td>Delete</td>
<td>978</td>
</tr>
<tr>
<td>Drop Table for Salesforce</td>
<td>978</td>
</tr>
<tr>
<td>Explain Plan</td>
<td>979</td>
</tr>
<tr>
<td>Insert</td>
<td>979</td>
</tr>
<tr>
<td>Select</td>
<td>981</td>
</tr>
<tr>
<td>Update</td>
<td>991</td>
</tr>
<tr>
<td>SQL Expressions</td>
<td>992</td>
</tr>
<tr>
<td>Subqueries</td>
<td>1000</td>
</tr>
<tr>
<td>Catalog tables</td>
<td>1002</td>
</tr>
<tr>
<td>SYSTEM_SESSIONS catalog table</td>
<td>1003</td>
</tr>
<tr>
<td>SYSTEM_REMOTE_SESSIONS catalog table</td>
<td>1004</td>
</tr>
<tr>
<td>Error messages</td>
<td>1006</td>
</tr>
<tr>
<td>Management API error messages</td>
<td>1006</td>
</tr>
<tr>
<td>Performance tuning</td>
<td>1025</td>
</tr>
<tr>
<td>Oracle Marketing Cloud bulk operations</td>
<td>1025</td>
</tr>
<tr>
<td><strong>Configuring an On-Premises Connector</strong></td>
<td>1029</td>
</tr>
<tr>
<td>Configuring the On-Premises Connector</td>
<td>1029</td>
</tr>
<tr>
<td>Restarting the On-Premises Connector</td>
<td>1030</td>
</tr>
<tr>
<td>Determining the Connector information</td>
<td>1030</td>
</tr>
<tr>
<td>Defining the proxy server</td>
<td>1031</td>
</tr>
<tr>
<td>Configuring On-Premises Connector memory resources</td>
<td>1032</td>
</tr>
<tr>
<td>Determining the version</td>
<td>1034</td>
</tr>
<tr>
<td>Checking the configuration status</td>
<td>1034</td>
</tr>
<tr>
<td>Configuring failover and balancing requests across multiple On-Premises Connectors</td>
<td>1034</td>
</tr>
<tr>
<td>Configuring the Microsoft Dynamics CRM On-Premises data source for Kerberos</td>
<td>1035</td>
</tr>
<tr>
<td>Troubleshooting the On-Premises Connector</td>
<td>1036</td>
</tr>
</tbody>
</table>
Hybrid Data Pipeline API reference..........................................................1037

Administrators API..........................................................................................................................1039
  Administrator Permissions API..............................................................................................................1039
Authentication API.................................................................................................................................1042
CORS Whitelist API................................................................................................................................1063
Limits API..................................................................................................................................................1071
Logging API...............................................................................................................................................1107
Roles API..................................................................................................................................................1112
System Configurations API......................................................................................................................1124
Tenant API..................................................................................................................................................1132
Users API..................................................................................................................................................1146
User Details API.......................................................................................................................................1191
Health Check API......................................................................................................................................1192
  Get health check......................................................................................................................................1193
  Head health check....................................................................................................................................1194
IP Address Whitelist API..........................................................................................................................1195
  Retrieve IP address whitelists at the system level.................................................................................1196
  Create IP address whitelists at the system level....................................................................................1199
  Update IP address whitelists at the system level....................................................................................1202
  Delete IP address whitelists at the system level.....................................................................................1205
  Retrieve tenants configured with IP address whitelists.........................................................................1205
  Retrieve IP address whitelists for a tenant..............................................................................................1208
  Create IP address whitelists for a tenant.................................................................................................1210
  Update IP address whitelists for a tenant...............................................................................................1213
  Delete IP address whitelists for a tenant...............................................................................................1216
  Retrieve users configured with whitelist IP............................................................................................1217
  Retrieve IP address whitelists for the authenticated user....................................................................1219
  Create IP address whitelists for the authenticated user......................................................................1221
  Update IP address whitelists for a user.................................................................................................1223
  Delete IP address whitelists for the authenticated user......................................................................1226
Management API......................................................................................................................................1227
  Connector API.........................................................................................................................................1227
  Data Sources API.................................................................................................................................1277
  Driver Files API....................................................................................................................................1360
  Management Permissions API...............................................................................................................1371
  OAuth API for configuring Hybrid Data Pipeline to authorize client applications..............................1373
  OAuth API for Google Analytics connectivity........................................................................................1391
  Schema API............................................................................................................................................1412
  Version Information API.......................................................................................................................1423
Password Policy API................................................................................................................................1424
  Get Password Policy............................................................................................................................1424
  Validate Password Policy....................................................................................................................1427
Hybrid Data Pipeline API Error Messages..............................................................................................1429
General errors........................................................................................................................1430
Administrator API error messages.........................................................................................1430
Connector API error messages..............................................................................................1433
Data Sources API error messages.........................................................................................1439
OAuth API error messages....................................................................................................1448
Welcome to DataDirect Hybrid Data Pipeline

Progress DataDirect Hybrid Data Pipeline is a light-weight software service that provides simple, secure access to cloud and on-premises data for business intelligence tools and applications. Client applications can use ODBC, JDBC, or OData to access data from over twenty supported relational and non-relational database management systems, such as Apache Hive, DB2, SQL Server, Oracle, and Salesforce (collectively referred to as data stores). Requests from client applications are translated into the format supported by the underlying data store – SQL, NoSQL, Big Data, cloud – and returned in the format accepted by the client. Communications in HTTP and HTTPS are supported.

Key Features

• Standards-based connectivity - Use ODBC, JDBC, or OData to access data stores hosted in the cloud and on-premises
• Multitenancy - Host multiple tenants from a single, physical instance of Hybrid Data Pipeline
• Cloud deployment - Host Hybrid Data Pipeline from the cloud for access to cloud and on-premises data sources without requiring a VPN or other gateway
• On-premises deployment - Host Hybrid Data Pipeline on-premises for access to cloud and on-premises data sources
• Load balancer deployment - Run the Hybrid Data Pipeline service on one or more nodes behind a load balancer to support scalability
• Standalone node deployment - Run the Hybrid Data Pipeline service on a single standalone node

Main Components

• Hybrid Data Pipeline server - Provides access to multiple data sources through a unified interface
• On-Premises Connector - Enables Hybrid Data Pipeline to establish a secure connection from the cloud to an on-premises data source without requiring a VPN or other gateway
• ODBC driver - Enables ODBC applications to communicate to a data source through the Hybrid Data Pipeline server (HTTP and HTTPS connections supported)

• JDBC driver - Enables JDBC applications to communicate to a data source through the Hybrid Data Pipeline server (HTTP and HTTPS connections supported)

For details, see the following topics:

• Product requirements
• Deployment guidelines
• Deployment scenarios

Product requirements

Hybrid Data Pipeline provides access to multiple data sources through a single, unified interface. The Hybrid Data Pipeline server can be supported with the installation and configuration of additional components such as the On-Premises Connector, the ODBC driver, and the JDBC driver.

Note: For REST-based data access for mobile apps and desktop applications, no local software is needed.

Before proceeding with the installation of the server or additional components, confirm that your environment meets the requirements described in the following sections.

• Hybrid Data Pipeline server
• On-Premises Connector
• JDBC driver
• ODBC driver
• Browser for Hybrid Data Pipeline Web UI

Hybrid Data Pipeline server

The Hybrid Data Pipeline server must be installed on a 64-bit Linux machine with, at minimum, 4 cores and 8 GB of RAM.
Note: The OpenJDK 8 JRE is installed with the server and used at runtime. However, you may integrate an external JRE to support the service. OpenJDK 8 and Oracle Java 8 JREs are supported for external integration. See External JRE support and integration on page 52 for details.

<table>
<thead>
<tr>
<th>Platform</th>
<th>64-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td>Operating System</td>
</tr>
<tr>
<td>• 4 core and 8 GB RAM minimum</td>
<td>• Centos 4, 5, 6, 7</td>
</tr>
<tr>
<td></td>
<td>• Oracle 4, 5, 6, 7</td>
</tr>
<tr>
<td></td>
<td>• Red Hat Enterprise 4, 5, 6, 7</td>
</tr>
<tr>
<td></td>
<td>• SUSE Enterprise Server 10, 11, 12, 13</td>
</tr>
<tr>
<td></td>
<td>• Ubuntu 16 and higher</td>
</tr>
</tbody>
</table>

**On-Premises Connector**

The On-Premises Connector must be installed on a 64-bit Windows machine with, at minimum, 4 cores and 8 GB of RAM.

Note: The OpenJDK 8 JRE is installed with the On-Premises Connector and used at runtime. However, you may integrate an external JRE to support the On-Premises Connector. OpenJDK 8 and Oracle Java 8 JREs are supported for external integration. See External JRE support and integration on page 52 for details.

<table>
<thead>
<tr>
<th>Platform</th>
<th>64-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>Operating System</td>
</tr>
<tr>
<td>• 4 core and 8 GB RAM minimum</td>
<td>• Windows 10</td>
</tr>
<tr>
<td></td>
<td>• Windows 8.1</td>
</tr>
<tr>
<td></td>
<td>• Windows 7</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2012 Service Pack 2</td>
</tr>
<tr>
<td></td>
<td>• Windows Server 2008</td>
</tr>
</tbody>
</table>

**JDBC driver**

An installation of the JDBC driver requires 21 MB of hard disk space at minimum. A supported JVM must be defined on your system path. The following JVM implementations are supported.

<table>
<thead>
<tr>
<th>JVM (32-bit and 64-bit JVMs supported)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Oracle Java 8 and 11</td>
</tr>
<tr>
<td>• OpenJDK 8 and 11</td>
</tr>
</tbody>
</table>
## ODBC driver

An installation of the ODBC driver requires 132 MB of hard disk space at minimum. The following platforms are supported.

<table>
<thead>
<tr>
<th>Platform</th>
<th>32-bit</th>
<th>64-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX</td>
<td>• 7.1</td>
<td>• 7.1</td>
</tr>
<tr>
<td></td>
<td>• 6.1</td>
<td>• 6.1</td>
</tr>
<tr>
<td></td>
<td>• 5.3 Fixpack 5</td>
<td>• 5.3 Fixpack 5 or higher</td>
</tr>
<tr>
<td>HP-UX PA-RISC</td>
<td>• 11i Version 3.0 (B.11.3x)</td>
<td>na</td>
</tr>
<tr>
<td></td>
<td>• 11i Version 2.0 (B.11.23)</td>
<td></td>
</tr>
<tr>
<td>HP-UX IPF</td>
<td>• 11i Version 3.0 (B.11.3x)</td>
<td>• 11i Version 3.0 (B.11.3x)</td>
</tr>
<tr>
<td></td>
<td>• 11i Version 2.0 (B.11.23)</td>
<td>• 11i Version 2.0 (B.11.23)</td>
</tr>
<tr>
<td>Linux</td>
<td>• CentOS Linux 4, 5, 6, 7</td>
<td>• CentOS Linux 4, 5, 6, 7</td>
</tr>
<tr>
<td></td>
<td>• Debian 7.11, 8.5</td>
<td>• Debian 7.11, 8.5</td>
</tr>
<tr>
<td></td>
<td>• Oracle Linux 4, 5, 6, 7</td>
<td>• Oracle Linux 4, 5, 6, 7</td>
</tr>
<tr>
<td></td>
<td>• Red Hat Enterprise 4, 5, 6, 7</td>
<td>• Red Hat Enterprise 4, 5, 6, 7</td>
</tr>
<tr>
<td></td>
<td>• SUSE Enterprise Server 10, 11, 12</td>
<td>• SUSE Enterprise Server 10, 11, 12</td>
</tr>
<tr>
<td></td>
<td>• Ubuntu 14.04, 16.04</td>
<td>• Ubuntu 14.04, 16.04</td>
</tr>
<tr>
<td>Oracle Solaris on SPARC</td>
<td>• Oracle Solaris 11, 11 Express</td>
<td>• Oracle Solaris 11, 11 Express</td>
</tr>
<tr>
<td></td>
<td>• Oracle Solaris 8, 9, 10</td>
<td>• Oracle Solaris 8, 9, 10</td>
</tr>
<tr>
<td>Oracle Solaris x86: Intel</td>
<td>• Oracle Solaris 11, 11 Express</td>
<td>na</td>
</tr>
<tr>
<td></td>
<td>• Oracle Solaris 10</td>
<td></td>
</tr>
</tbody>
</table>
## Deployment guidelines

Hybrid Data Pipeline is a highly adaptable software service that can be securely integrated into a variety of network environments. Follow these guidelines to get your Hybrid Data Pipeline environment up and running.

- Determine how to deploy Hybrid Data Pipeline. Hybrid Data Pipeline can be deployed either on a standalone node or behind a load balancer on one or more nodes. See Deployment scenarios on page 22 for detailed information.

- Determine which components you need to install and configure in addition to the Hybrid Data Pipeline server. The ODBC driver must be installed to support ODBC applications, and the JDBC driver to support JDBC applications. The On-Premises Connector can be installed for direct, secure access to on-premises data sources.

- Ensure that Product requirements on page 18 are met for each component you are installing. At this time, the Hybrid Data Pipeline server must be installed on a Linux 64-bit machine.

### Browser for Hybrid Data Pipeline Web UI

The following browsers are supported.

<table>
<thead>
<tr>
<th>Browser</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>Chrome 53.0 and higher</td>
</tr>
<tr>
<td>Edge</td>
<td>42 and higher</td>
</tr>
<tr>
<td>Firefox</td>
<td>Firefox 48 and higher</td>
</tr>
<tr>
<td>Internet Explorer</td>
<td>Internet Explorer 11.0 and higher</td>
</tr>
<tr>
<td>Safari</td>
<td>Safari 9.1 and higher</td>
</tr>
</tbody>
</table>
• Collect the information needed for server installation. For example, host and port information must be supplied during the installation of the Hybrid Data Pipeline server. The information you need, in part, depends on your deployment scenario.

• Install the Hybrid Data Pipeline server. Refer to the Progress DataDirect Hybrid Data Pipeline Installation Guide for details.

• After installation of at least one Hybrid Data Pipeline server, you can modify your environment to use an external JRE at runtime as opposed to the embedded JRE that is shipped with the product package. See External JRE support and integration on page 52.

• After the installation of the server, proceed with the installation of supporting components, such as the On-Premises Connector, as appropriate. Refer to the Progress DataDirect Hybrid Data Pipeline Installation Guide for details.

• Build out the Hybrid Data Pipeline environment. See Administering Hybrid Data Pipeline on page 57.
  • Establish a single-tenant or multitenant architecture. See Tenant architectures on page 82.
  • Use the Web UI or Administrators API to provision users. See User provisioning on page 107 for details.
  • Use the Web UI or Data Sources API to create data sources to support queries to data stores such as Oracle and Salesforce. See Creating data sources with the Web UI on page 224 and Data Sources API on page 1277.

• Configure your OData applications to query the data sources you have created. See Configuring data sources for OData connectivity and working with data source groups on page 622, Getting started with OData Version 4 on page 863, and Getting started with OData Version 2 on page 827.

• Configure the ODBC and JDBC drivers, as well as your ODBC and JDBC applications, to query data sources. See Getting started with the ODBC Driver on page 650 and Getting started with the JDBC driver on page 731.

Deployment scenarios

Hybrid Data Pipeline can be deployed on a standalone machine or on one or more nodes behind a load balancer. Many configurations and best practices are contingent on how Hybrid Data Pipeline has been deployed.

For a production environment, Hybrid Data Pipeline should be deployed on one or more nodes behind a load balancer to support scalability and availability. In a load balancer deployment, client application requests must be directed to the load balancer which forwards requests to the node or nodes running the service. When multiple nodes have been deployed, requests are distributed across the cluster. See Load balancer deployment on page 36 for more information.

When deployed on a standalone node, the service is installed on a single host machine that manages all queries, simplifying maintenance and administration. A standalone deployment is not recommended for a production environment because it does not provide the scalability and availability of a load balancer deployment. However, a standalone deployment may be required due to resource limitations and other restrictions. If a standalone deployment is required in production, then, as a matter of best practices, the deployment should include an external system database and a user-specified key location. See Standalone deployment on page 23 for details.
**Important:** There is currently no migration path from a standalone deployment to a load balancer deployment. Therefore, a standalone deployment is not recommended for environments where scaling up the service may be desired. A standalone node deployment is also not recommended for security and system recovery purposes. If you want to move from a test environment to a production environment, you should begin by deploying Hybrid Data Pipeline on a single node behind a load balancer. When deploying the service on a single node behind a load balancer, you can increase availability and scalability as demanded, and address security and recovery concerns as required.

Whether you deploy the service on a standalone node or behind a load balancer, Hybrid Data Pipeline can be run on-premises or in the cloud. See the following topics for more information.

- Exposing on-premises data sources to cloud-based applications on page 51
- Connecting an application in the cloud to on-premises data sources on page 52.

In addition, after at least one installation of the Hybrid Data Pipeline server, you can modify your environment to use an external JRE at runtime as opposed to the embedded JRE that is shipped with the product package. See External JRE support and integration on page 52.

### Standalone deployment

Hybrid Data Pipeline configuration depends in part on whether you are deploying the service on a standalone node or deploying the service on one or more nodes behind a load balancer. The standalone deployment simplifies installation and administration of the service. For this reason, the standalone deployment is an efficient way to test proof of concepts and evaluate the service. In a standalone deployment, the service is installed on a single host machine and queries must be directed to this machine.

Hybrid Data Pipeline is largely configured during the installation process. The following configuration details should be addressed before installation to ensure a successful standalone deployment.

- **Login credentials for standalone deployment** on page 24
  
  Passwords for the default administrator and user accounts must be specified during installation of the Hybrid Data Pipeline server. When initially logging in to the Web UI or using the API, you must authenticate as one of these users.

- **System database for standalone deployment** on page 24
  
  A system database is required for storing user and configuration information. For standalone deployments, you can use either the embedded internal database or a supported external database to serve as the system database. However, an external system database should be used in production environments.

- **Shared files and the key location for standalone deployment** on page 28
  
  The installation program creates shared files used in the operation of the data access service. During installation, you choose where and how these files should be stored. In a production environment, the files used to connect to the system database should be secured on a machine separate from the machines hosting the Hybrid Data Pipeline service and the system database. In addition, all shared files should be backed up as a matter of best practices. In the case of system failure, these backups can be used to restore the service.

- **Access ports for standalone deployment** on page 29
  
  The access ports used for Hybrid Data Pipeline should be enabled for incoming traffic and unallocated for other purposes.

- **SSL certificates for standalone deployment** on page 30
To implement SSL/TLS in a Hybrid Data Pipeline environment, an SSL certificate file must be specified during installation. For standalone deployments, a self-signed certificate is available for testing or evaluation purposes, but a PEM file should be specified to enable SSL in a production environment.

- **Application and driver configuration for standalone deployment** on page 34
  Applications and drivers must be properly configured to ensure a successful deployment of the service.

- **Firewall and port redirection using iptables for standalone deployment** on page 35
  Hybrid Data Pipeline Web UI and API endpoints are exposed by default on port 8080 for HTTP connections or port 8443 for HTTPS connections. The iptables firewall utility can be used to route connections from the standard HTTP port 80 and HTTPS port 443 to these endpoints.

### Login credentials for standalone deployment

You must specify passwords for the default *d2cadmin* and *d2cuser* accounts during installation of the Hybrid Data Pipeline server. The default password policy is not enforced during installation of the server. However, best practices recommend that you follow the default password policy when specifying these account passwords. When initially logging in to the Web UI or using Hybrid Data Pipeline APIs, you must authenticate as one of these users.

#### Hybrid Data Pipeline default password policy

After installation, Hybrid Data Pipeline enforces the following password policy by default.

- The password must contain at least 8 characters.
- The password must not contain more than 12 characters. A password with a length of 12 characters is acceptable.
- The password must not contain the username.
- Characters from at least three of the following four groups must be used in the password:
  - Uppercase letters A-Z
  - Lowercase letters a-z
  - Numbers 0-9
  - Non-white space special characters

### System database for standalone deployment

Hybrid Data Pipeline requires a system database for storing user and configuration information. When deploying the service on a standalone node, you can opt to use either the embedded internal database or a supported external database. A standalone installation with an internal system database is the quickest way to get Hybrid Data Pipeline up and running. With this deployment, the service can be installed and administered from a single machine. This deployment is an efficient way to test and evaluate the service. However, for a production environment, an external system database should be used. An external system database provides better security and more flexibility for backing up system information. As a best practice, the external system database should be replicated, or mirrored, to promote the continuous availability of the service. Configuring Hybrid Data Pipeline to use a system database occurs during installation.
**External system databases**

Hybrid Data Pipeline requires a system database for storing sensitive information used in the operation of the data access service. For a standalone node deployment, you can opt to use either the embedded internal database or a supported external database. For a load balancer deployment, you must use an external database. Depending on the external database you are using, certain requirements must be met. See the following sections for details.

- [Supported databases](#) on page 25
- [Oracle requirements](#)
- [MySQL Community Edition requirements](#) on page 26
- [Microsoft SQL Server requirements](#) on page 27
- [PostgreSQL requirements](#) on page 27

### Supported databases

**Note:** Hybrid Data Pipeline supports Amazon RDS instances that are compatible with these supported database versions.

<table>
<thead>
<tr>
<th>Database</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Azure SQL Database</td>
<td>Microsoft Azure SQL Database 11</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>Microsoft SQL Server 2016</td>
</tr>
<tr>
<td></td>
<td>Microsoft SQL Server 2014</td>
</tr>
<tr>
<td>MySQL Community Edition</td>
<td>Support based on MySQL Connector/J 5.1&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>Oracle Database</td>
<td>Oracle 12c R1, R2 (12.1, 12.2)</td>
</tr>
<tr>
<td></td>
<td>Oracle 11g R2 (11.2)</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>PostgreSQL 11</td>
</tr>
</tbody>
</table>

<sup>1</sup> Hybrid Data Pipeline does not provide a driver for MySQL Community Edition. MySQL Connector/J 5.1 must be used to support the use of MySQL Community Edition as an external system database. Therefore, you should refer to the MySQL Connector/J 5.1 documentation for information on supported versions of MySQL Community Edition.
Oracle requirements
If you plan to store system information in an external Oracle database, you must provide the following information.

- Hostname (server name or IP address)
- Port information for the database. The default is 1521.
- SID or Service Name
- Administrator and user account information
  - An administrator name and password. The administrator must have the following privileges:
    - CREATE SESSION
    - CREATE TABLE
    - CREATE ANY SYNONYM
    - CREATE SEQUENCE
    - CREATE TRIGGER
  - A user name and password for a standard account. The standard user must have the CREATE SESSION privileges.

MySQL Community Edition requirements
If you plan to use a MySQL Community Edition database as an external system database, you must provide the following.

- A MySQL Connector/J driver, version 5.1, and its location
  To download the driver, visit the MySQL developer website at https://dev.mysql.com/.
- Hostname (server name or IP address)
- Port information for the database. The default is 3306.
- Database Name
- Administrator and user account information:
  - An administrator user name and password. The administrator must have the following privileges:
    - ALTER
    - CREATE
    - DROP
    - DELETE
    - INDEX
    - INSERT
    - REFERENCES
    - SELECT
    - UPDATE
  - A user name and password for a standard account. The standard user must have the following privileges:
    - DELETE
Microsoft SQL Server requirements

If you plan to store system information in an external SQL Server database, you must take the following steps when setting up the SQL Server database.

1. Create a database schema to be used for storing Hybrid Data Pipeline system information.
2. Create an administrator who can access the newly created schema. The administrator must have the CREATE TABLE privileges.
3. Create a user who can access the newly created schema. The user must have the CREATE SESSION privileges.

After the SQL Server database has been setup, you must provide the following information during installation:

- Hostname (server name or IP address)
- Port information for the database. The default is 1433.
- Database Name
- Schema Name
- Administrator and user account information
  - An administrator name and password. The administrator must have the CREATE TABLE privileges.
  - A user name and password for a standard account. The user must have the CREATE SESSION privileges.

PostgreSQL requirements

If you plan to store system information on an external PostgreSQL database, you must take the following steps when setting up the PostgreSQL database.

1. Enable the citext PostgreSQL extension.
2. Create a database schema to be used for storing Hybrid Data Pipeline system information.
3. Create an administrator who can access the newly created schema. The administrator must have privileges to create tables.
4. Create a user who can access the newly created schema. The user must have privileges to select, insert, update, delete, and sequence tables.

After the PostgreSQL database has been setup, you must provide the following information during installation:

- Hostname (server name or IP address)
- Port information for the database. The default is 5432.
- Database Name
- Administrator and user account information
  - An administrator name and password. The administrator must have privileges to create tables.
  - A user name and password for a standard account. The user must have privileges to select, insert, update, delete, and sequence tables.
Shared files and the key location for standalone deployment

Hybrid Data Pipeline requires the specification of a key location during installation. For a standalone deployment, if you use the default key location, the installation program writes the shared files used in the operation of the data access service to the local keystore directory (<install_dir>/ddcloud/keystore). If you specify a different location as the key location, the installation program writes the shared files to two separate locations. The files necessary for connecting to the system database are stored in the specified location, while files tied to the Hybrid Data Pipeline server are stored in the local keystore directory (<install_dir>/ddcloud/keystore).

In a production environment, the files used to connect to the system database should be secured on a machine separate from the machines hosting the Hybrid Data Pipeline service and the system database. Therefore, a separate location should be specified for the key location.

Whether located in a single directory or two separate directories, all shared files should be backed up as a matter of best practices. In the case of system failure, these backups can be used to restore the service.

Note: During installation of the Hybrid Data Pipeline server, four configuration and certificate files are generated. These files are used in the installation of components, including the ODBC driver, the JDBC driver, and the On-Premises Connector. In a standalone node installation, the location of these files is independent of the shared location. These files are written to the <install_dir>/redist directory.

Shared files

The following files are used to connect to the system database. When the default location is used for the key location, these files are stored in the local keystore directory (<install_dir>/ddcloud/keystore). When a non-default location is used, these files are stored in the location specified during installation.

- .backup: A backup copy of the contents of the install directory from the previous install. This is used to restore the contents of the directory if there is an error during an upgrade.
- key: Reference to the file containing the encryption key for the Hybrid Data Pipeline database.
- key00: Encryption key for the system database. This key is used to encrypt sensitive information such as data source user IDs and passwords, security tokens, access tokens and other user or data source identifying information. If this is not present, or was over written during the installation, then you will not be able decrypt any of the encrypted information in the system database.
- key-cred: Encryption key for credentials contained in Hybrid Data Pipeline configuration files. Examples of credentials in the config files include the user ID and password information for the system database.
- db/*: Encrypted information about the system database. The contents of these files are encrypted using the key-cred key. Used by the installer when performing an upgrade or installing on an additional node. If these are not present, or do not have valid encoding, the installation or upgrade will fail.
- dddrivers/*: A directory of internally supported drivers that have been updated after a product upgrade.
- drivers/*: The directory used for integrating third party drivers with Hybrid Data Pipeline.
- plugins/*: JAR files for external authentication plugins

The following files are tied to the Hybrid Data Pipeline server. They are stored in the local keystore directory (<install_dir>/ddcloud/keystore) whether or not the default key location is specified during installation.

- authKey: Authentication key for the On-Premises Connector. This key is used to encrypt the user ID and password information in the On-Premises Connector configuration file. The key in this file is encrypted using a key built into the On-Premises Connector. This encrypted key is included in the OnPremise.properties configuration file distributed with the On-Premises Connector. If this is overwritten or incorrect, the On-Premises Connector will not be able to authenticate with Hybrid Data Pipeline.
• **ddcloud.jks**: Sun SSL keystore. This keystore contains the Hybrid Data Pipeline server SSL certificate if the SSL termination is done at the Hybrid Data Pipeline server.

• **ddcloud.bks**: Bouncy Castle SSL keystore. This keystore contains the same SSL certificate as the **ddcloud.jks** keystore. This keystore is in the Bouncy Castle keystore format and is used when the server is configured to run in FIPS compliant mode. Should only be present with FIPS enabled.

• **ddcloudTrustStore.jks**: Sun SSL truststore. This truststore contains the root CA certificate needed to validate the server SSL certificate. This truststore is distributed with the On-Premises Connector and with the ODBC and JDBC drivers, allowing these components to validate the Hybrid Data Pipeline server certificate.

• **ddcloudTrustStore.bks**: Bouncy Castle SSL truststore. Should only be present with FIPS enabled. This truststore contains the root CA certificate needed to validate the server SSL certificate in the Bouncy Castle keystore format. The Bouncy Castle SSL library does not use the default Java cacerts file, so this truststore is populated with the contents of the default cacerts file and the root certificate needed to validate the Hybrid Data Pipeline server certificate. Should only be present with FIPS enabled.

• **key-opc**: Contains the unencrypted encryption key. The authKey above contains the encrypted version of this key. This key is not shipped with the On-Premises Connector.

### Access ports for standalone deployment

Multiple access ports on the machine hosting the Hybrid Data Pipeline server must be opened and unassigned to other functions. The following tables document the required ports and default port numbers for standalone deployments. The installation program for the Hybrid Data Pipeline server confirms that default ports are available and allows new port values to be assigned when needed. Port values are passed during the installation of Hybrid Data Pipeline servers.

#### Server Access Port

For a standalone installation, a Server Access Port must be available across the firewall of the Hybrid Data Pipeline server. Using an HTTPS port is recommended.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Port</td>
<td>8080</td>
<td>Port that exposes Hybrid Data Pipeline</td>
</tr>
<tr>
<td>HTTPS Port</td>
<td>8443</td>
<td>SSL port that exposes Hybrid Data Pipeline</td>
</tr>
</tbody>
</table>

#### Server Internal Ports

The Shutdown Port must be opened. However, as a matter of best practice, the Shutdown Port should not be available outside the firewall of the Hybrid Data Pipeline server. For a standalone node installation, a port for the Internal API must be opened. Using the Internal API SSL Port is recommended.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal API Port</td>
<td>8190</td>
<td>Non-SSL port for the Internal API</td>
</tr>
<tr>
<td>Internal API SSL Port</td>
<td>8090</td>
<td>SSL port for the Internal API</td>
</tr>
<tr>
<td>Shutdown Port</td>
<td>8005</td>
<td>Shutdown port</td>
</tr>
</tbody>
</table>
On-Premises Access Ports
The Message Queue Port must be opened. For a standalone node installation with the On-Premises Connector, the On-Premises Access Port and a Notification Server Port must be available across the firewall. Using the SSL port is recommended.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Premises Port</td>
<td>40501</td>
<td>Port for the On-Premises Connector</td>
</tr>
<tr>
<td>TCP Port</td>
<td>11280</td>
<td>Port for the Notification Server</td>
</tr>
<tr>
<td>SSL Port</td>
<td>11443</td>
<td>SSL port for the Notification Server</td>
</tr>
<tr>
<td>Message Queue Port</td>
<td>8282</td>
<td>Port for the message queue</td>
</tr>
</tbody>
</table>

SSL certificates for standalone deployment
To implement SSL/TLS in a Hybrid Data Pipeline environment, an SSL certificate file must be specified during installation. In a standalone deployment, the Hybrid Data Pipeline server needs a server certificate and all intermediate certificates all the way to the root of the certificate chain to establish trust. During installation, you can specify a self-signed certificate for testing or evaluation purposes. However, as documented below, a PEM file should be specified to enable SSL in a production environment.

Note: The ODBC driver, JDBC driver, and On-Premises Connector need only the root certificate to verify the trust of the server certificate supplied during the SSL handshake. During installation of the server, the required certificate files are written to the <install dir>/redist directory. These and other files in the redist directory must be used in the installation of the ODBC driver, JDBC driver, and On-Premises Connector.

An SSL/TLS implementation secures the following communications in a standalone deployment.

- Communications between a Hybrid Data Pipeline user and the Hybrid Data Pipeline Web UI.
- Communications between applications using the REST API, including the OData API, and the Hybrid Data Pipeline server.
- Communications between the JDBC or ODBC drivers and the Hybrid Data Pipeline server.
- Communications between the On-Premises Connector and the Hybrid Data Pipeline server.

The PEM file
To implement SSL/TLS, a standalone Hybrid Data Pipeline deployment should be configured with a server certificate issued by a certificate authority. For a client to verify the authenticity of a certificate, it needs to be able to verify the signatures of all of the certificates in the chain. As such, the entire certificate chain must be supplied when configuring the Hybrid Data Pipeline server including the root certificate.

A PEM file must consist of a private key, a CA server certificate, and additional certificates that make up the trust chain. The trust chain must contain a root certificate and, if needed, intermediate certificates.

A PEM encoded file includes Base64 data. The private key is prefixed with a "-----BEGIN PRIVATE KEY-----" line and postfixed with an "-----END PRIVATE KEY-----". Certificates are prefixed with a "-----BEGIN CERTIFICATE-----" line and postfixed with an "-----END CERTIFICATE-----" line. Text outside the prefix and postfix lines is ignored and can be used for metadata.
You may need to create a PEM file by converting different key and certificate files into separate PEM files, and then concatenating these files into a single PEM file. In some cases, you may need to first convert key and certificate files into a PKCS12 (pfx) file and then convert the PKCS12 file into a PEM file. The PEM file should include the private key and required certificates, as shown in PEM file format on page 31.

PEM file format

A PEM file must consist of a private key, a CA server certificate, and additional certificates that make up the trust chain. The trust chain must contain a root certificate and, if needed, intermediate certificates.

A PEM encoded file includes Base64 data. The private key is prefixed with a "-----BEGIN PRIVATE KEY-----" line and postfixed with an "-----END PRIVATE KEY-----". Certificates are prefixed with a "-----BEGIN CERTIFICATE-----" line and postfixed with an "-----END CERTIFICATE-----" line. Text outside the prefix and postfix lines is ignored and can be used for metadata.

The following example PEM file contains a private key, a CA server certificate, one intermediate trust chain certificate, and a root certificate. The ellipses (...) represent additional lines of text in the certificate or key that have been removed.

```
# Private key
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBgwgSkAgEAcoIBAQDB08s5++4anG
cmKQJXlABgkVNBa0THVyb2dy2XNaIFNz2r3YXJ1IENYcnBvcmF0aw9uMSAwhYD
VQQDDB9cmlF3cy10ZXN0LmByb2dy2XNzLmNvbTCCAS1wDQYJKoZIhvcNAQEBBQAD
... 
bm16YXxrb252YWxaGEyZzIuY3JMlGiBGbrBgEFQEQgBAQSBKbzCBkDNBBggrBgEF
BQcwaAoZBHRcDoVlL3N1Yi3Vj2S5nbG91YWxzaWduLmNvbS9jYW1nckVz3Nvcmdh
z3P668Yf9DkmKfR6S42Cg6zn
-----END PRIVATE KEY-----

# Server CA certificate
-----BEGIN CERTIFICATE-----
MIIFaDCCBFCgAwIBAgISESHkvZFwK9Qz0RksX0g1BgNVBAoTHVByb2dy2XNaIFNz2r3YXJ1
IENYcnBvcmF0aw9uMSAwhYDVQQDDB9cmlF3cy10ZXN0LmByb2dy2XNzLmNvbTCCAS1w
DQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAMGPTyynn77hcqYnjWswOZDzdhVFY93s2O
JntMbuKTHn39B... 
bm16YXxrb252YWxaGEyZzIuY3JMlGiBGbrBgEFQEQgBAQSBKbzCBkDNBBggrBgEF
BQcwaAoZBHRcDoVlL3N1Yi3Vj2S5nbG91YWxzaWduLmNvbS9jYW1nckVz3Nvcmdh
bm16YXxrb252YWxaGEyZzIuY3JMlGiBGbrBgEFQEQgBAQSBKbzCBkDNBBggrBgEF
BQcwaAoZBHRcDoVlL3N1Yi3Vj2S5nbG91YWxzaWduLmNvbS9jYW1nckVz3Nvcmdh
-----END CERTIFICATE-----

# Trust chain intermediate certificate
-----BEGIN CERTIFICATE-----
MIIEaTCCA1GgAwIBAgILBAAAAABRE7wQzgwDQYJKoZIhvcNAQELBQAwVzELMAKGC
C3Jj1J1fD/4ncxMYTXbVz6xvJvYb2dxKrkt3Ia2z12vBq9QO95FyDLzmn+ebomchj
Snb/Vz2Pghk1jWLUdCok1H/hgKBueq16lfYygo0OhJc0izeg0k9zfrH0SwU... 
dHbix0i8vd3dLmdsb2ZjbbhNpa2Zz4uY29lL3Jl1cG9zaXVckvMDMGAlUdHwQQMCow
KKAmoCSGMh0dHA6ly91cmwuZXZvYmFsc21nb5u2ZQXvcn9vcD5cmwwPQYIKwYB
K1p7Y4P18S8qtCr4fKxhZS9Ma9yHDPSsQPhZ2s=-----END CERTIFICATE-----

# Trust chain root certificate
-----BEGIN CERTIFICATE-----
MIIDfTCCCA12gAwIBAgILBAAAAABRFtaw5QowDQYJKoZIhvcNAQEEFBgwVzELMAKGC
YXNzaWduIG52LXNhMRAwDgYDVQQLEw52LXNhMRMwEQYDVQQDEw52LXNhMRMwEQYDVQQD
... 
-----END CERTIFICATE-----
```

Deployment scenarios
See also
The PEM file on page 30
Generating a PEM file on page 32

Generating a PEM file

A PEM file must consist of a private key, a CA server certificate, and additional certificates that make up the trust chain. The trust chain must contain a root certificate and, if needed, intermediate certificates.

You may need to create a PEM file by converting different key and certificate files into separate PEM files, and then concatenating these files into a single PEM file. In some cases, you may need to first convert key and certificate files into a PKCS12 file and then convert the PKCS12 file into a PEM file. The resulting PEM file should include the private key and required certificates, as shown in PEM file format on page 31.

The following sections describe a number of ways to convert key and certificate files, using OpenSSL or the Java keytool as appropriate.

- Converting a PKCS12 (pfx) file to a PEM file on page 32
- Converting a Java jks keystore file to a PKCS12 file on page 33
- Converting PKCS7 (p7b) file certificates to PEM file certificates on page 33
- Converting PKCS7 file certificates to PKCS12 file certificates and adding the private key to the PKCS12 file on page 33
- Converting DER certificates to PEM file certificates on page 34
- Creating a PEM file from a private key and Base64 encoded certificates on page 34

Converting a PKCS12 (pfx) file to a PEM file

1. Use the following OpenSSL command to determine whether the private key is password protected.

   openssl pkcs12 -info -in target.pfx

   a. If the key is password protected, you will be prompted for a password. Proceed to Step 2.
   b. If the key is not password protected, then information on the PKCS12 file, such as file structure and algorithms used, is provided. Proceed to Step 5.

2. Enter the password when prompted. Information on the PKCS12 file, such as file structure and algorithms used, is provided.

3. Use the following OpenSSL command to extract the private key from the PKCS12 file.

   openssl pkcs12 -in target.pfx -nocerts -out ppkey.pem

4. Remove the passphrase from the private key. Then, skip to Step 6.

   openssl rsa -in ppkey.pem -out privatekey.pem

5. Use the following OpenSSL command to extract the private key from the PKCS12 file.

   openssl pkcs12 -in target.pfx -nocerts -out privatekey.pem

6. Extract the root certificates from the PKCS12 file.

   openssl pkcs12 -in rootcert.pfx -cacerts -nodes -nokeys > rootcert.pem
7. Extract server certificates from the PKCS12 file.

   openssl pkcs12 -in servercert.pfx -clcerts -nodes -nokeys > servercert.pem

8. Concatenate the certificates and private key in a single PEM file. In this example, the Linux/UNIX `cat` command is used to concatenate root certificate, server certificate, and private key.

   ```
cat rootcert.pem servercert.pem privatekey.pem > server.bundle.pem
   ```

9. Confirm that the PEM file has the private key and the required certificates as described in PEM file format on page 31.

   The resulting `server.bundle.pem` file should be specified during the installation of the Hybrid Data Pipeline server.

Converting a Java jks keystore file to a PKCS12 file

A Java jks keystore file must first be converted to a PKCS12 file. The PKCS12 file can then be converted to a PEM file.

1. Use the following Java keytool command to convert the jks file into a pfx file.

   ```
   keytool -importkeystore -srckeystore keystore.jks -srcstoretype JKS -deststoretype PKCS12 -destkeystore target.pfx
   ```

2. Enter the keystore password and keystore file alias when prompted.

3. Use the resulting `target.pfx` file to create a PEM file by following the instructions in Converting a PKCS12 (pfx) file to a PEM file on page 32.

Converting PKCS7 (p7b) file certificates to PEM file certificates

These instructions assume that the private key is already available as a PEM file.

1. Use the following OpenSSL command to convert PKCS7 file certificates to PEM file certificates.

   ```
   openssl pkcs7 -print_certs -in certificates.p7b -out certificates.pem
   ```

2. Concatenate the certificate and private key files. In this example, the Linux/UNIX `cat` command is used.

   ```
cat certificates.pem privatekey.pem > server.bundle.pem
   ```

3. Confirm that the resulting PEM file has the private key and the required certificates as described in PEM file format on page 31.

   The resulting `server.bundle.pem` file should be specified during the installation of the Hybrid Data Pipeline server.

Converting PKCS7 file certificates to PKCS12 file certificates and adding the private key to the PKCS12 file

After the certificate and private key files have been converted to the PKCS12 format, the PKCS12 file can then be converted to a PEM file.
1. Use the following OpenSSL command to convert a PKCS7 file to a PKCS12 file.

   ```
   openssl pkcs7 -print_certs -in certificate.p7b -out certificate.cer
   ```

2. Use the following command to add the private key to the PKCS12 file.

   ```
   openssl pkcs12 -export -in certificate.cer -inkey privatekey.key -out target.pfx
   -certfile CACert.cer
   ```

3. Use the resulting `target.pfx` file to create a PEM file by following the instructions in Converting a PKCS12 (pfx) file to a PEM file on page 32.

Converting DER certificates to PEM file certificates

The DER extension is used for binary DER files. These files may also use the CER and CRT extensions. These instructions assume that the private key is already available as a PEM file.

1. Use the following OpenSSL command to convert DER certificates to PEM file certificates.

   ```
   openssl x509 -inform der -in certificates.cer -out certificates.pem
   ```

2. Concatenate the certificate and private key files. In this example, the Linux/UNIX `cat` command is used.

   ```
   cat certificates.pem privatekey.pem > server.bundle.pem
   ```

3. Confirm that the PEM file has the private key and the required certificates as described in PEM file format on page 31.

   The resulting `server.bundle.pem` file should be specified during the installation of the Hybrid Data Pipeline server.

Creating a PEM file from a private key and Base64 encoded certificates

PEM files use Base64 encoding. Therefore, no conversion process is required. However, the Base64 encoded certificates and the private key must be concatenated in a single PEM file.

These instructions assume that the private key is already available as a PEM file.

1. Concatenate the certificate and private key files. In this example, the Linux/UNIX `cat` command is used.

   ```
   cat Base64rootcert.pem Base64servercert.pem privatekey.pem > server.bundle.pem
   ```

2. Confirm that the PEM file has the private key and the required certificates as described in PEM file format on page 31.

   The resulting `server.bundle.pem` file should be specified during the installation of the Hybrid Data Pipeline server.

See also

- The PEM file on page 30
- PEM file format on page 31

Application and driver configuration for standalone deployment

Client applications must be appropriately configured. In conjunction with ODBC and JDBC applications, ODBC and JDBC drivers will also need to be configured. OData applications will need their own modifications.
For the most part, configuration of the ODBC and JDBC drivers is handled during the installation of the drivers. If the drivers are installed using the configuration files generated by the Hybrid Data Pipeline server installation, then they will use the DNS of the host machine. Nevertheless, you may wish to configure the drivers in other ways.

OData applications must be modified to use the DNS of the host machine for HTTP or HTTPS requests. In addition, OData applications should be configured for SSL as appropriate.

**Firewall and port redirection using iptables for standalone deployment**

Hybrid Data Pipeline Web UI and API endpoints are exposed by default on port 8080 for HTTP connections or port 8443 for HTTPS connections. The iptables firewall utility can be used to route connections from the standard HTTP port 80 and HTTPS port 443 to these endpoints. In this scenario, ports 80 and 443 will be accessible to everyone, while ports 8080 and 8443 are only accessible to processing running on the server.

The instructions in the following topics can be applied to RedHat 7, Oracle 7 and Centos 7 distributions of Linux.

Please see the documentation for your Linux distribution for more information about configuring the firewall.

---

**Note:** If you are using a Suse 12 distribution of Linux, use the YaST2 Firewall setting GUI to configure your firewall. In Suse 12 you can find the firewall setting under Applications > System Tools > YaST > Administrator Settings/Security and Users/Firewall.

---

**Disabling firewalld**

If you are using a later version of Linux, it may have come configured with the newer firewalld software. Consult the documentation for firewalld to determine how to configure it in a similar way, and how to disable firewalld and use iptables.

To disable firewalld, use the following commands in a console window.

```
systemctl disable firewalld
systemctl stop firewalld
```

**Installing iptables**

Installing iptable requires root privileges.

1. Log in with an admin account.
2. Run `sudo -s`
3. Use `yum` to install the iptables services:
   - a) `yum install iptables`
   - b) `yum install iptables-ipv6`
Creating the iptables configuration file

Create the file /etc/sysconfig/iptables containing the content displayed here (your configuration may be slightly different). This will require root privileges.

```
# Generated by iptables-save v1.4.21 on Thu Jun 23 09:05:43 2016
*nat
:PREROUTING ACCEPT [1100:133346]
:INPUT ACCEPT [1:48]
:OUTPUT ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
-A PREROUTING -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 8080
-A PREROUTING -p tcp -m tcp --dport 443 -j REDIRECT --to-ports 8443
-A PREROUTING -p tcp --dport 8080 -j MARK --set-mark 1
-A PREROUTING -p tcp --dport 8443 -j MARK --set-mark 2
COMMIT
# Completed on Thu Jun 23 09:05:43 2016
# Generated by iptables-save v1.4.21 on Thu Jun 23 09:05:43 2016
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [378:34583]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -m mark --mark 1 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 8080 -j ACCEPT
-A INPUT -m mark --mark 2 -j DROP
-A INPUT -p tcp -m state --state NEW -m tcp --dport 8443 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 443 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Thu Jun 23 09:05:43 2016
```

Starting the iptables service

Start the iptables service using the `service` command.

```
service iptables start
```

Load balancer deployment

Hybrid Data Pipeline configuration depends in part on whether you are deploying the service on a standalone node or deploying the service on one or more nodes behind a load balancer. A load balancer deployment offers high availability and scalability, and is therefore the best option for production environments. In a load balancer deployment, the service is installed on one or more nodes behind a load balancer. Requests are handled by the load balancer which distributes requests across nodes.

Hybrid Data Pipeline is largely configured during the installation process. When installing the service on multiple nodes behind a load balancer, the initial installation of the Hybrid Data Pipeline server is used as a template for installations on additional nodes. The following configuration details should be addressed before installation to ensure a successful load balancer deployment.

- **Login credentials for load balancer deployment** on page 37

  Passwords for the default administrator and user accounts must be specified during installation of the Hybrid Data Pipeline server. When initially logging in to the Web UI or using the API, you must authenticate as one of these users.
• **Load balancer configuration** on page 38
  Hybrid Data Pipeline can be deployed on one or more nodes behind a load balancer to provide high availability and scalability. Hybrid Data Pipeline supports two types of load balancers.
  
  • **Network load balancers that support the TCP tunneling protocol (such as HAProxy)**
  • **Cloud load balancers that support the WebSocket protocol (such as the AWS application load balancer and the Azure application gateway)**

• **System database for load balancer deployment** on page 44
  A system database is required for storing user and configuration information. For load balancer deployments, an external database is required to serve as the system database. As a best practice, the external system database should be replicated, or mirrored, to promote the continuous availability of the service.

• **Shared files and the key location for load balancer deployment** on page 48
  The specification of a *key location* is required during installation. The installation program writes shared files used in the operation of the data access service to this directory. As a matter of best practices, the key location should be secured on a machine separate from the machines hosting the Hybrid Data Pipeline service or the machine hosting the system database.

• **Access ports for load balancer deployment** on page 49
  The access ports used for Hybrid Data Pipeline should be enabled for incoming traffic and unallocated for other purposes.

• **SSL certificates for load balancer deployment** on page 49
  SSL/TLS encrypted communications between client applications and the load balancer are supported. In addition, all communications between the On-Premises Connector and the load balancer are SSL/TLS encrypted. SSL connections between the load balancer and the Hybrid Data Pipeline nodes are currently not supported.

• **Client application configuration for load balancer deployment** on page 50
  Applications and drivers must be properly configured to ensure a successful deployment of the service.

• **Browser configuration for load balancer deployment** on page 51
  For load balancer deployments, the browser you use to connect to the Web UI must have cookies enabled.

**Login credentials for load balancer deployment**

You must specify passwords for the default *d2cadmin* and *d2cuser* accounts during installation of the Hybrid Data Pipeline server. The default password policy is not enforced during installation of the server. However, best practices recommend that you follow the default password policy when specifying these account passwords.

When initially logging in to the Web UI or using Hybrid Data Pipeline APIs, you must authenticate as one of these users.

**Hybrid Data Pipeline default password policy**

After installation, Hybrid Data Pipeline enforces the following password policy by default.

• The password must contain at least 8 characters.

• The password must not contain more than 12 characters. A password with a length of 12 characters is acceptable.

• The password must not contain the username.
Characters from at least three of the following four groups must be used in the password:

- Uppercase letters A-Z
- Lowercase letters a-z
- Numbers 0-9
- Non-white space special characters

**Load balancer configuration**

The Hybrid Data Pipeline product package does not include a load balancer. However, Hybrid Data Pipeline can be deployed on one or more nodes behind a load balancer to provide high availability and scalability. Hybrid Data Pipeline supports two types of load balancers: network load balancers that support the TCP tunneling protocol and cloud load balancers that support the WebSocket protocol. In turn, the load balancer must be configured to support the Hybrid Data Pipeline environment according to the following criteria.

- The load balancer must be configured to accept HTTPS connections on port 443 and unencrypted HTTP connections on port 80.
- The load balancer must be configured for SSL termination to support encrypted communications between clients and the load balancer. The configuration of the load balancer depends in part on the type of SSL certificate supplied. See SSL certificates for load balancer deployment on page 49 for details.
- The load balancer must support session affinity. The load balancer must either be configured to supply its own cookies or to pass the cookies generated by the Hybrid Data Pipeline service back to the client. The Hybrid Data Pipeline service provides a cookie named C2S-SESSION that can be used by the load balancer. For ODBC and JDBC applications, the ODBC and JDBC drivers automatically use cookies for session affinity. OData applications should be configured to echo cookies for optimal performance.
- The load balancer must pass the hostname in the Host header when a request is made to an individual Hybrid Data Pipeline node. For example, if the hostname used to access the cluster is hdp.mycorp.com and the individual nodes behind the load balancer have the hostnames hdpsvr1.mycorp.com, hdpsvr2.mycorp.com, hdpsvr3.mycorp.com, then the Host header in the request forwarded to the Hybrid Data Pipeline node must be the load balancer hostname hdp.mycorp.com.
- The load balancer must supply the X-Forwarded-Proto header to indicate to the Hybrid Data Pipeline node whether the request was received by the load balancer as an HTTP or HTTPS request.
- The load balancer must supply the X-Forwarded-For header for IP address filtering. The X-Forwarded-For header is also required if the client IP address is needed for Hybrid Data Pipeline access logs. If the X-Forwarded-For header is not supplied, the IP address in the access logs will always be the load balancer’s IP address.
- The load balancer may be configured to run HTTP health checks against nodes with the Health Check API.
- Additional configuration is required for the following scenarios.
  - If you are using the On-Premises Connector with a network load balancer such as HAProxy, see Configuring a network load balancer with the On-Premises Connector on page 39 for additional configuration requirements.
  - If you are using the On-Premises Connector with a cloud load balancer such as the AWS Application Load Balancer or the Azure Application Gateway, see Configuring a cloud load balancer with the On-Premises Connector on page 42 for additional configuration details.
Configuring a network load balancer with the On-Premises Connector

When running Hybrid Data Pipeline behind a network load balancer with an On-Premises Connector, the load balancer must be configured to route requests for on-premises data sources to the correct server nodes.

There are two general steps involved in configuring your load balancer to support on-premises data access. First, a custom Access Control List must be created to direct requests for the On-Premises Connector to cluster nodes. Second, a backend notification pool that specifies the on-premises port for each cluster node must be created. The following instructions explain how an HAProxy load balancer can be configured to support Hybrid Data Pipeline access to backend data sources using the On-Premises Connector. These instructions may be adapted for other load balancers, such as NGINX and F5.

The Hybrid Data Pipeline installation program automatically generates an HAProxy configuration file for each installation of the server. These HAProxy configuration files are written to the HAProxy subdirectory in the key location directory specified during installation. These files must be merged to create a single HAProxy configuration file for a load balancer deployment of Hybrid Data Pipeline.

Take the following steps to create an HAProxy configuration file for a load balancer deployment using the On-Premises Connector.

1. Create an Access Control List (ACL) to direct requests for the On-Premises Connector to each Hybrid Data Pipeline server.

   **Note:** Options 1 and 2 below may be used in combination.

   **Option 1.** Use a custom header to direct requests. Each entry should be prefaced with acl.
   
   In this example, the custom header X-DataDirect-OPC-Host is used to direct requests to the server service2.myserver.com through the default On-Premises Port 40501.
   
   ```
   acl is_opa_hdr_service2_myserver_com_40501 hdr(X-DataDirect-OPC-Host)
   -i opa_service2_myserver_com_40501
   use_backend opa_service2_myserver_com_40501 if is_opa_hdr_service2_myserver_com_40501
   ```

   **Option 2.** Use URL routing to direct requests. Each entry should be prefaced with acl.
   
   In this example, URL routing is used to direct requests to the server service2.myserver.com through the default On-Premises Port 40501.
   
   ```
   acl is_opa_url_service2_myserver_com_40501 path_end
   -i /connect/opa_service2_myserver_com_40501
   use_backend opa_service2_myserver_com_40501 if is_opa_url_service2_myserver_com_40501
   ```

2. Add each Hybrid Data Pipeline server to the backend notification pool section using the server keyword.

   In the following example, the server server2.myserver.com has been added to the backend hdp_notification_pool section, and health checks have been enabled at the root with the option httpchk property.
   
   ```
   backend hdp_notification_pool
   mode http
   option http-tunnel
   balance roundrobin
   option httpchk HEAD /
   http-check expect status 200
   #HDP Notification Server Definitions
   server server1.myserver.com 11.22.111.105:11280 check
   server server2.myserver.com 11.22.111.106:11280 check
   ```
3. Create a backend pool that specifies the On-Premises Port for each Hybrid Data Pipeline server that supports the On-Premises Connector by adding a backend section to the configuration file.

For example, the following backend section is for a node on the service2.myserver.com server using the default On-Premises Port 40501. Health checks have been enabled at the root with the option httpchk property.

```plaintext
backend opa_service2_myserver_com_40501
  mode http
  option http-tunnel
  option httpchk HEAD / http-check expect status 200
  server service2.myserver.com 11.22.111.106:40501 check
```

4. Add each Hybrid Data Pipeline server to the default backend pool using the server keyword.

In the following example, server2.myserver.com has been added to the backend hdp_default_backend pool, and health checks have been enabled by specifying the /api/healthcheck endpoint with the option httpchk property.

```plaintext
backend hdp_default_backend
  mode http
  balance roundrobin
  option httpchk HEAD /api/healthcheck http-check expect status 200
  cookie HDP_SESSION insert nocache

  #HDP Server Definitions
  server service1.myserver.com 11.22.110.8080 check cookie service1.myserver.com
  server service2.myserver.com 11.22.111.106:8080 check cookie service2.myserver.com
```

Example

The following example demonstrates an HAProxy configuration file for using the load balancer with two server nodes that have the On-Premises connector enabled, service1.myserver.com and service2.myserver.com. To create this file, the required sections were copied from the generated configuration file for service2.myserver.com into the generated file for service1.myserver.com. Copied sections are indicated with comments.

```plaintext
global
  log 127.0.0.1 local0
  chroot /var/lib/haproxy
  daemon

defaults
  log   global
  mode  http
  option httplog
  option dontlognull
  timeout connect 5s
  timeout client 15m
  timeout server 15m

# Configuration for OPC with load balancer.
#----------------------------------------------------------------------
frontend lb_opc_nodes
  bind *:80
  # Replace /common/hdpsmoke/shared/redist/ddcloud.pem with the location of the
  # load balancers SSL certificate
  bind *:443 ssl crt /common/hdpsmoke/shared/redist/ddcloud.pem

  # In production port 80 should be a permanent redirected to 443 by uncommenting the
```
#following line
#redirect scheme https code 301 if !{ ssl_fc }

mode http
default_backend hdp_default_backend

#Define rules for HDP Notification Servers
acl is_hdp_notification2 path_end -i /connect/X_DataDirect_Notification_Server
use_backend hdp_notification_pool if is_hdp_notification2

acl is_hdp_notification hdr(X-DataDirect-OPC-Host) -i X_DataDirect_Notification_Server
use_backend hdp_notification_pool if is_hdp_notification

#Rules for on-premises connection to service.myserver.com
acl is_url_opa_service1_myserver_com_40501 path_end
-i /connect/opa_service1_myserver_com_40501
use_backend opa_service1_myserver_com_40501 if is_url_opa_service1_myserver_com_40501

acl is_hdr_opa_service1_myserver_com_40501 hdr(X-DataDirect-OPC-Host)
-i opa_service1_myserver_com_40501
use_backend opa_service1_myserver_com_40501 if is_hdr_opa_service1_myserver_com_40501

backend hdp_notification_pool
  mode http
  option http-tunnel
  balance roundrobin
  option httpchk HEAD /
  http-check expect status 200

#HDP Notification Server Definitions
server service1.myserver.com 11.22.111.105:11280 check
#The following server argument was copied from the service2.myserver.com
#configuration file
server service2.myserver.com 11.22.111.106:11280 check

backend opa_service1_myserver_com_40501
  mode http
  option http-tunnel
  option httpchk HEAD /
  http-check expect status 200
  server service1.myserver.com 11.22.111.105:40501 check

#The following section was copied from the service2.myserver.com configuration file.
backend opa_service2_myserver_com_40501
  mode http
  option http-tunnel
  option httpchk HEAD /
  http-check expect status 200
  server service2.myserver.com 11.22.111.106:40501 check

backend hdp_default_backend
  mode http
Configuring a cloud load balancer with the On-Premises Connector

Hybrid Data Pipeline can be deployed on a web service, such as Amazon Web Services or Microsoft Azure, behind a cloud load balancer that supports the WebSocket protocol. When using an On-Premises Connector, the cloud load balancer must be configured to route requests for on-premises data sources to the correct server nodes.

The instructions in this section describe how an Amazon Web Services load balancer must be configured to support Hybrid Data Pipeline. These instructions assume that you have completed the following deployment tasks.

• Created a Virtual Private Cloud (VPC) to host a Hybrid Data Pipeline environment.
• Created AWS compute instances in the VPC for each node that will be used to support the Hybrid Data Pipeline environment.
• Provisioned an RDS database instance to operate as a system database for storing user and configuration information.
• Created a file system on a node in the VPC to be used as the key location for shared files.
• Installed the Hybrid Data Pipeline server on each node that will be hosting the service.
  • The key location specified during the initial installation must reside on a node in the VPC.
  • The system database specified during initial installation must be the RDS database instance for storing user and configuration information.
• Created an AWS Application Load Balancer in the VPC to connect to Hybrid Data Pipeline.

The following general steps must be taken to configure routing and listening rules in the AWS Application Load Balancer. The corresponding topics provide detailed instruction for each step.

1. Create a target group for default routing to the Hybrid Data Pipeline service API on page 42
2. Create a target group for notifications on page 43
3. Create a target group for on-premises access on page 43
4. Configure target routing on page 44

Once the Application Load Balancer has been configured with listener and target group rules, you can install On-Premises Connectors.

Create a target group for default routing to the Hybrid Data Pipeline service API

Take the following steps to create a target group for default routing.
1. Use the AWS console to create a load balancer target group.

2. Specify target group details.

   Name - <Name for your HDP cluster nodes>
   Protocol - HTTP
   Port 8080
   Target type - Instance
   VPC <Name of your VPC>

3. Set up health checks.

   Protocol: HTTP
   Port: 8080
   Path: /api/healthcheck

4. Save the target group.

5. Register each Hybrid Data Pipeline instance as a target on port 8080.

6. Set the stickiness attribute for the target group to 5 minutes.

**Create a target group for notifications**

Take the following steps to create a target group for notifications.

1. Use the AWS console to create a load balancer target group.

2. Specify target group details.

   Target Group Name: <Name for your Notification Server Group>
   Protocol HTTP
   Port 11280
   Target type instance
   VPC <Name of your VPC>

3. Set up health checks.

   Protocol: HTTP
   Path: /
   Port: Select traffic port

4. Save the target group.

5. Register each Hybrid Data Pipeline instance as a target on port 11280.

6. Disable stickiness via the stickiness attribute.

**Create a target group for on-premises access**

Take the following steps to create a target group for on-premises access.

1. Use the AWS console to create a load balancer target group.

2. Specify target group details.

   Target Group Name: <Name for your 1st OPA Target Group>
   Protocol HTTP
   Port 40501
   Target type instance
   VPC <Name of your VPC>
3. Set up health checks.

   Protocol: HTTP
   Path: /
   Port: Select traffic port

4. Save the target group.

5. Register the first Hybrid Data Pipeline instance as a target on port 40501.

6. Disable stickiness via the stickiness attribute.

7. Repeat steps 1 through 6 for each Hybrid Data Pipeline instance.

**Configure target routing**

Take the following steps to configure target routing.

1. Create a rule to route to the notifications target group by setting Path is to /connect/X_DataDirect_Notification_Server.

   **Note:** For load balancers that support routing with HTTP headers, the header X-DataDirect-OPC-Host:X_DataDirect_Notification_Server should be used.

2. For each node running the Hybrid Data Pipeline service, create a rule to route to the corresponding on-premises access target by setting Path is to /connect/<opa_routing_key>.

   **Note:** The format of the <opa_routing_key> is opa_<hosturl>_<opaport> where <hosturl> is the hostname specified during installation with dot characters replaced by underscores, and <opaport> is the On-Premises Access port number. For example, the routing key for nc-d2c02.americas.test.com on port 40501 would be opa_nc-d2c73_americas_test_com_40501.

3. Create a default routing rule. The Forward to attribute should be set to the Hybrid Data Pipeline service API target group.

   **Important:** Setting the default rule for routing requests to the Hybrid Data Pipeline service API must be completed after creating the rules for routing to the On-Premises Access and Notifications servers.

**System database for load balancer deployment**

Hybrid Data Pipeline requires a system database for storing user and configuration information. When deploying the service behind a load balancer, you must use a supported external database. An external system database ensures that user and configuration information is consistent across multiple nodes behind the load balancer. These nodes use the system information on the external system database to access data and return successful queries. In addition, an external system database provides better security and more flexibility for backing up system information. As a best practice, the external system database should be replicated, or mirrored, to promote the continuous availability of the service. Configuring Hybrid Data Pipeline to use a system database occurs during installation.
External system databases

Hybrid Data Pipeline requires a system database for storing sensitive information used in the operation of the data access service. For a standalone node deployment, you can opt to use either the embedded internal database or a supported external database. For a load balancer deployment, you must use an external database. Depending on the external database you are using, certain requirements must be met. See the following sections for details.

- **Supported databases** on page 45
- **Oracle requirements**
- **MySQL Community Edition requirements** on page 46
- **Microsoft SQL Server requirements** on page 47
- **PostgreSQL requirements** on page 47

Supported databases

**Note:** Hybrid Data Pipeline supports Amazon RDS instances that are compatible with these supported database versions.

<table>
<thead>
<tr>
<th>Database</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Azure SQL Database</td>
<td>Microsoft Azure SQL Database 11</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>Microsoft SQL Server 2016</td>
</tr>
<tr>
<td></td>
<td>Microsoft SQL Server 2014</td>
</tr>
<tr>
<td>MySQL Community Edition</td>
<td>Support based on MySQL Connector/J 5.1¹</td>
</tr>
<tr>
<td>Oracle Database</td>
<td>Oracle 12c R1, R2 (12.1, 12.2)</td>
</tr>
<tr>
<td></td>
<td>Oracle 11g R2 (11.2)</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>PostgreSQL 11</td>
</tr>
</tbody>
</table>

¹ Hybrid Data Pipeline does not provide a driver for MySQL Community Edition. MySQL Connector/J 5.1 must be used to support the use of MySQL Community Edition as an external system database. Therefore, you should refer to the MySQL Connector/J 5.1 documentation for information on supported versions of MySQL Community Edition.
Oracle requirements

If you plan to store system information in an external Oracle database, you must provide the following information.

- Hostname (server name or IP address)
- Port information for the database. The default is 1521.
- SID or Service Name
- Administrator and user account information
  - An administrator name and password. The administrator must have the following privileges:
    - CREATE SESSION
    - CREATE TABLE
    - CREATE ANY SYNONYM
    - CREATE SEQUENCE
    - CREATE TRIGGER
  - A user name and password for a standard account. The standard user must have the CREATE SESSION privileges.

MySQL Community Edition requirements

If you plan to use a MySQL Community Edition database as an external system database, you must provide the following.

- A MySQL Connector/J driver, version 5.1, and its location
  - To download the driver, visit the MySQL developer website at https://dev.mysql.com/.
- Hostname (server name or IP address)
- Port information for the database. The default is 3306.
- Database Name
- Administrator and user account information:
  - An administrator user name and password. The administrator must have the following privileges:
    - ALTER
    - CREATE
    - DROP
    - DELETE
    - INDEX
    - INSERT
    - REFERENCES
    - SELECT
    - UPDATE
  - A user name and password for a standard account. The standard user must have the following privileges:
    - DELETE
Microsoft SQL Server requirements

If you plan to store system information in an external SQL Server database, you must take the following steps when setting up the SQL Server database.

1. Create a database schema to be used for storing Hybrid Data Pipeline system information.
2. Create an administrator who can access the newly created schema. The administrator must have the CREATE TABLE privileges.
3. Create a user who can access the newly created schema. The user must have the CREATE SESSION privileges.

After the SQL Server database has been setup, you must provide the following information during installation:

- Hostname (server name or IP address)
- Port information for the database. The default is 1433.
- Database Name
- Schema Name
- Administrator and user account information
  - An administrator name and password. The administrator must have the CREATE TABLE privileges.
  - A user name and password for a standard account. The user must have the CREATE SESSION privileges.

PostgreSQL requirements

If you plan to store system information on an external PostgreSQL database, you must take the following steps when setting up the PostgreSQL database.

1. Enable the citext PostgreSQL extension.
2. Create a database schema to be used for storing Hybrid Data Pipeline system information.
3. Create an administrator who can access the newly created schema. The administrator must have privileges to create tables.
4. Create a user who can access the newly created schema. The user must have privileges to select, insert, update, delete, and sequence tables.

After the PostgreSQL database has been setup, you must provide the following information during installation:

- Hostname (server name or IP address)
- Port information for the database. The default is 5432.
- Database Name
- Administrator and user account information
  - An administrator name and password. The administrator must have privileges to create tables.
  - A user name and password for a standard account. The user must have privileges to select, insert, update, delete, and sequence tables.
Shared files and the key location for load balancer deployment

Hybrid Data Pipeline requires the specification of a *key location* during installation. The installation program writes shared files used in the operation of the data access service to this directory. For a load balancer deployment, the key location must be accessible to the node or nodes running the service.

Shared files

The following files are stored in the key location for a load balancer deployment.

- **.backup**: A backup copy of the contents of the install directory from the previous install. This is used to restore the contents of the directory if there is an error during an upgrade.
- **key**: Reference to the file containing the encryption key for the Hybrid Data Pipeline database.
- **key00**: Encryption key for the system database. This key is used to encrypt sensitive information such as data source user IDs and passwords, security tokens, access tokens and other user or data source identifying information. If this is not present, or was over written during the installation, then you will not be able decrypt any of the encrypted information in the system database.
- **key-cred**: Encryption key for credentials contained in Hybrid Data Pipeline configuration files. Examples of credentials in the config files include the user ID and password information for the system database.
- **db/**: Encrypted information about the system database. The contents of these files are encrypted using the **key-cred** key. Used by the installer when performing an upgrade or installing on an additional node. If these are not present, or do not have valid encoding, the installation or upgrade will fail.
- **dddrivers/**: A directory of internally supported drivers that have been updated after a product upgrade.
- **drivers/**: The directory used for integrating third party drivers with Hybrid Data Pipeline.
- **plugins/**: JAR files for external authentication plugins.
- **authKey**: Authentication key for the On-Premises Connector. This key is used to encrypt the user ID and password information in the On-Premises Connector configuration file. The key in this file is encrypted using a key built into the On-Premises Connector. This encrypted key is included in the OnPremise.properties configuration file distributed with the On-Premises Connector. If this is overwritten or incorrect, the On-Premises Connector will not be able to authenticate with Hybrid Data Pipeline.
- **ddcloud.jks**: Sun SSL keystore. This keystore contains the Hybrid Data Pipeline server SSL certificate if the SSL termination is done at the Hybrid Data Pipeline server.
- **ddcloud.bks**: Bouncy Castle SSL keystore. This keystore contains the same SSL certificate as the ddcloud.jks keystore. This keystore is in the Bouncy Castle keystore format and is used when the server is configured to run in FIPS compliant mode. Should only be present with FIPS enabled.
- **ddcloudTrustStore.jks**: Sun SSL truststore. This truststore contains the root CA certificate needed to validate the server SSL certificate. This truststore is distributed with the On-Premises Connector and with the ODBC and JDBC drivers, allowing these components to validate the Hybrid Data Pipeline server certificate.
- **ddcloudTrustStore.bks**: Bouncy Castle SSL truststore. Should only be present with FIPS enabled. This truststore contains the root CA certificate needed to validate the server SSL certificate in the Bouncy Castle keystore format. The Bouncy Castle SSL library does not use the default Java cacerts file, so this truststore is populated with the contents of the default cacerts file and the root certificate needed to validate the Hybrid Data Pipeline server certificate. Should only be present with FIPS enabled.
- **key-opc**: Contains the unencrypted encryption key. The **authKey** above contains the encrypted version of this key. This key is not shipped with the On-Premises Connector.
- **global.properties**: Stores properties and other information shared between nodes in a cluster.
• `redist/*`: Redistributable files. These files are used to install the On-Premises Connector and the ODBC and JDBC drivers.

**Access ports for load balancer deployment**

Multiple access ports on nodes hosting the Hybrid Data Pipeline server must be opened and unassigned to other functions. The following tables document the required ports and default port numbers. The installation program for the Hybrid Data Pipeline server confirms that default ports are available and allows new port values to be assigned when needed. Port values are passed during the installation of Hybrid Data Pipeline servers.

**Server Access Port**

A Server Access Port must be opened for the load balancer. As a matter of best practices, the load balancer should be configured for SSL/TLS termination.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Port</td>
<td>8080</td>
<td>Port that exposes Hybrid Data Pipeline</td>
</tr>
</tbody>
</table>

**Server Internal Ports**

The Shutdown Port must be opened. However, as a matter of best practice, the Shutdown Port should not be available outside the firewall of the Hybrid Data Pipeline server. For a load balancer installation, the Internal API Port on any node must be open to all the other nodes in the cluster. The Internal API Port should not be available outside the firewall.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal API Port</td>
<td>8190</td>
<td>Non-SSL port for the Internal API</td>
</tr>
<tr>
<td>Shutdown Port</td>
<td>8005</td>
<td>Shutdown port</td>
</tr>
</tbody>
</table>

**On-Premises Access Ports**

The Message Queue Port must be opened. For a load balancer installation with the On-Premises Connector, the On-Premises Access Port and the TCP Notification Server Port must be opened for the load balancer.

<table>
<thead>
<tr>
<th>Name</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Premises Port</td>
<td>40501</td>
<td>Port for the On-Premises Connector</td>
</tr>
<tr>
<td>TCP Port</td>
<td>11280</td>
<td>Port for the Notification Server</td>
</tr>
<tr>
<td>Message Queue Port</td>
<td>8282</td>
<td>Port for the message queue</td>
</tr>
</tbody>
</table>

**SSL certificates for load balancer deployment**

The following SSL encrypted communications are supported for a load balancer deployment.

- Communications between the browser and the Hybrid Data Pipeline Web UI when the load balancer is configured for SSL.
- Communications between applications using the REST API, including the OData API, and the load balancer.
• Communications between the JDBC or ODBC drivers and the load balancer.
• Communications between the On-Premises Connector and the load balancer.

**Important:** SSL connections between the load balancer and the Hybrid Data Pipeline nodes are currently not supported.

The following guidelines should be used when implementing SSL in a Hybrid Data Pipeline environment.

- The load balancer needs to be configured with the root certificate and any intermediate certificates necessary to establish the chain of trust to the root certificate.
- The root certificate must be specified as the SSL certificate during installation of the Hybrid Data Pipeline server. When intermediate certificates are required for the trust chain, then the SSL certificate must be supplied in a PEM file format. When there are no intermediate certificates, then the SSL certificate can be supplied in either DER or PEM file format.
- The SSL certificate specified during installation is used to generate the trust stores for the ODBC driver, JDBC driver, and On-Premises Connector. These files are written to the `redist` directory of the key location upon installation. Before installing the ODBC driver, the JDBC driver, or the On-Premises Connector, the trust store and properties files in the `redist` directory must be copied to the installer directory of the component you are installing.

**Client application configuration for load balancer deployment**

Client applications must be appropriately configured. In conjunction with ODBC and JDBC applications, ODBC and JDBC drivers will also need to be configured. OData applications will need their own modifications.

For the most part, configuration of the ODBC and JDBC drivers is handled during the installation of the drivers. If the drivers are installed using the configuration files generated by the Hybrid Data Pipeline server installation, then they will use the hostname of the load balancer or machine hosting the server. However, you may wish to configure the drivers in other ways.

OData applications must be modified to use the hostname of the load balancer for HTTP or HTTPS requests. Additionally, for optimal performance, OData applications should be configured to echo cookies for session affinity. OData applications must also be configured appropriately for SSL. See Node-to-node communication in OData Hybrid Data Pipeline load balancer environment on page 50 for details on communication between nodes when an OData client cannot be configured to echo cookies.

**Node-to-node communication in OData Hybrid Data Pipeline load balancer environment**

In an OData Hybrid Data Pipeline load balancer environment, the load balancer and OData clients should be configured to handle cookies to achieve session affinity and optimize OData query performance. The load balancer should supply its own cookies or pass the cookies generated by the Hybrid Data Pipeline service back to the OData client. In turn, the OData client should echo cookies to allow the load balancer to direct query requests to the node that initially received the query.

However, it is not always possible to configure an OData client to echo cookies. In such cases, Hybrid Data Pipeline uses an internal mechanism called the distributed file persistence manager. When a query is executed that requires file persistence, execution results are stored temporarily on the node that initially received the query. The manager associates the query with the node and the execution results stored there. If a request from the same query is routed to a different node in the cluster, the manager obtains the persisted execution results from the original node. The query results are then returned to the client by the node that received the request.
The distributed file persistence manager requires node-to-node communication using the HTTP protocol to achieve session affinity. The Internal API Port specified during Hybrid Data Pipeline server installation is the port used for this node-to-node communication. Data remains secure in the following respects. First, the Internal API Port (8190 default) is not exposed externally to the public facing network. Each node registers itself using this port, and communications are restricted. Second, a UUID is generated during the node registration process. This UUID is passed in as an HTTP header to confirm the validity of node-to-node communications. Third, the service stores persisted files on only a temporary basis.

### Browser configuration for load balancer deployment

For load balancer deployments of Hybrid Data Pipeline, the browser you use to connect to the Web UI must have cookies enabled.

### Exposing on-premises data sources to cloud-based applications

This scenario describes a deployment where on-premises data sources are exposed for secure access by cloud-based applications. For this deployment, a Hybrid Data Pipeline server is installed in the cloud, and the On-Premises Connector is used to perform secure connections through the firewall to the backend data store. The cloud-based application is located in a separate cloud but connects with Hybrid Data Pipeline through an API such as OData, ODBC, or JDBC.

This deployment could be suitable for an independent software vendor who wants to embed Hybrid Data Pipeline services in the cloud to give the cloud application users access to their data that resides in the data center or other on-premises systems.

For a more detailed discussion of this scenario, watch a video. 🎥
Connecting an application in the cloud to on-premises data sources

This scenario describes a deployment where the Hybrid Data Pipeline server is installed behind a firewall with on-premises data sources while a number of applications reside in the cloud. With the Hybrid Data Pipeline server behind a firewall, a cloud-based service does not need to be maintained, and SSL can be used to secure your data.

This deployment scenario could be suitable when using cloud-based OData applications, for example, creating a real-time connectivity between Salesforce and an on-premises database.

For a more detailed discussion of this scenario, watch a video.

External JRE support and integration

Hybrid Data Pipeline uses an embedded JRE at runtime. However, you can integrate an external JRE with a standing deployment of Hybrid Data Pipeline. The following JREs are currently supported.

- Oracle Java 8 JRE
- OpenJDK 8 JRE

Hybrid Data Pipeline must be installed on at least one server before you proceed with integrating an external JRE. Files associated with the embedded JRE can then be used to modify the external JRE you wish to use with the Hybrid Data Pipeline server or the On-Premises Connector.

Note: Using an external JRE with the server is exclusive from using an external JRE with the On-Premises Connector. That is, the server can run on an external JRE while the On-Premises Connector runs on the embedded JRE, and vice versa.

The following work flow outlines the procedure for integrating an external JRE. See the corresponding topics for details.

1. Modify the external JRE.
• **Option 1.** Non-FIPS environment.

• **Option 2.** FIPS environment.

**Note:** FIPS is not supported for the On-Premises Connector with either embedded or external JREs.

2. If integrating the external JRE with the server, **configure the server to use the JRE.**

3. If integrating the external JRE with the On-Premises Connector, **configure the connector to use the JRE.**

### Modify the external JRE for a non-FIPS environment

Take the following steps to modify an external JRE for a non-FIPS environment.

**Note:**

- `<hdp_install_dir>` is the installation directory of the Hybrid Data Pipeline server.
- `<external_jre_home>` is the home directory of the external JRE.

1. Enable the Unlimited Strength Jurisdiction Policy according to the JRE vendor documentation. Depending on the vendor and version, the Unlimited Strength Jurisdiction Policy may be enabled by default.

   **Note:** Enabling the Unlimited Strength Jurisdiction Policy is the only modification required for using an external JRE with the On-Premises Connector. Therefore, the remaining steps can be ignored if the JRE is to be used only with the On-Premises Connector.

2. Copy the `<hdp_install_dir>/ddcloud/utils/jre/lib/ext/bc-fips-1.0.0.jar` file to the `<external_jre_home>/lib/ext` directory.

3. Merge the contents of the embedded JRE

   **Note:**
   
   - Any previously made customizations to the `<external_jre_home>/lib/security/java.policy` should be preserved.
   
   - Any permissions for data sources in the embedded JRE `java.policy.sun` file should be carried over to the external JRE `java.policy` file.

4. Merge the contents of the embedded JRE

   **Note:**
   
   - Any previously made customizations to the `<external_jre_home>/lib/security/java.security` should be preserved.
   
   - Any properties enabled in the embedded JRE `java.security.sun` file should be carried over to the external JRE `java.security` file.

**What to do next:**

- Configure the server to use the external JRE.
• Configure the On-Premises Connector to use the external JRE.

**Modify the external JRE for a FIPS environment**

Take the following steps to modify an external JRE for a FIPS environment.

**Note:** FIPS is not supported for the On-Premises Connector with either embedded or external JREs.

**Note:**

- `<hdp_install_dir>` is the installation directory of the Hybrid Data Pipeline server.
- `<external_jre_home>` is the home directory of the external JRE.

1. Enable the Unlimited Strength Jurisdiction Policy according to the JRE vendor documentation. Depending on the vendor and version, the Unlimited Strength Jurisdiction Policy may be enabled by default.

2. Copy the `<hdp_install_dir>/ddcloud/utils/jre/lib/ext/bc-fips-1.0.0.jar` file to the `<external_jre_home>/lib/ext` directory.


   **Note:**

   - Any previously made customizations to the `<external_jre_home>/lib/security/java.policy` file should be preserved.
   - Any permissions for data sources in the embedded JRE `java.policy.bcfips` file should be carried over to the external JRE `java.policy` file.


   **Note:**

   - Any previously made customizations to the `<external_jre_home>/lib/security/java.security` file should be preserved.
   - Any properties enabled in the embedded JRE `java.security.bcfips` file should be carried over to the external JRE `java.security` file.

**What to do next:**

Configure the server to use the external JRE.

**Configure the server to use the external JRE**

Once you have modified the external JRE, you can configure the server to use the external JRE by performing an upgrade installation of the server. During the upgrade, you will be prompted to specify whether you are using the embedded JRE or an external JRE. If you select external JRE, you must specify the path to the external JRE.

**Note:** For complete upgrade instructions, refer to the **Progress DataDirect Hybrid Data Pipeline Installation Guide**.
If you are using a response file to perform a silent upgrade, best practices recommend that you use the installation program to generate the response file. However, you may opt to edit the response file manually. If editing the response file manually, you must add Java configuration options to the response file. The options and values depend on whether the response file is based on the GUI installation template or the console mode installation template.

**GUI mode**

```markdown
#Java Configuration#
------------------
SPECIFY_JAVA_HOME_NO=0
SPECIFY_JAVA_HOME_YES=1
HDP_JAVA_HOME_DIR=/usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.181-3.b13.el7_5.x86_64
```

- **SPECIFY_JAVA_HOME_NO** indicates whether you are using an external JRE. If you are using an external JRE, specify 0.
- **SPECIFY_JAVA_HOME_YES** indicates whether you are using an external JRE. If you are using an external JRE, specify 1.
- **HDP_JAVA_HOME_DIR** specifies the path to the external JRE to be used at runtime.

**Console mode**

```markdown
#Java Configuration#
------------------
SPECIFY_JAVA_HOME_YESNO="Yes","
HDP_JAVA_HOME_DIR_CONSOLE="/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.102-4.b14.el7.x86_64/jre"
```

**Important**: The escape characters, as shown in this example, are required for a response file based on the console mode template.

- **SPECIFY_JAVA_HOME_YESNO** indicates whether you are using an external JRE. If you are using an external JRE, specify Yes.
- **HDP_JAVA_HOME_DIR_CONSOLE** specifies the path to the external JRE to be used at runtime.

**What to do next:**

If integrating the external JRE with the On-Premises Connector, configure the connector to use the JRE.

**Configure the On-Premises Connector to use the external JRE**

To use an external JRE with an On-Premises Connector, the JRE’s Unlimited Strength Jurisdiction Policy must be enabled. No other modifications to the JRE are required to use it with an On-Premises Connector. Depending on the vendor and version of the JRE, the Unlimited Strength Jurisdiction Policy may be enabled by default.

Once the Unlimited Strength Jurisdiction Policy has been enabled, you can configure the On-Premises Connector to use the external JRE when installing or upgrading the connector. During installation or upgrade, you will be prompted to specify whether you are using the embedded JRE or an external JRE. If you select external JRE, you must specify the path to the external JRE. For complete installation instructions, refer to the Progress DataDirect Hybrid Data Pipeline Installation Guide.
Administering Hybrid Data Pipeline

The administration of Hybrid Data Pipeline involves the management of three basic resources common to any Hybrid Data Pipeline environment: tenants, user accounts, and data sources.

A Hybrid Data Pipeline system administrator can develop either a single-tenant or multitenant architecture. In a single-tenant architecture, the system administrator creates user accounts in the default system tenant. In a multitenant architecture, the system administrator first creates one or more child tenants in the system tenant. Then, the system administrator may create user accounts in either the system tenant or any one of the child tenants. The user accounts that reside in one tenant are isolated from those in other tenants.

Once a tenant architecture has been established, a system administrator can provision user accounts in two general ways. First, an administrator can provision an account such that the user has direct access to the Hybrid Data Pipeline service. In this case, the administrator can provision the user to create, manage, and query data sources. The administrator can also promote or restrict access to Hybrid Data Pipeline features, such as the Web UI, the SQL Editor, and the Management API.

Alternatively, an administrator can provision an account such that user access is limited to queries against a data source. For example, an administrator may want to provision user access such that a user can run an OData application against a backend data store. In this scenario, the administrator creates the data source and supplies the end user with connection information for the data source. The end user can query the data store with the connection information supplied, but he or she does not have access to the connection information stored in the data source definition or to Hybrid Data Pipeline itself.

Beginning with information on initial log in, the topics in this section provide information on administering Hybrid Data Pipeline and configuring Hybrid Data Pipeline features.

For details, see the following topics:

- Initial login with default user accounts
- Permissions and default roles
- Logging in to the Web UI
Chapter 2: Administering Hybrid Data Pipeline

- Using Hybrid Data Pipeline APIs
- Using the Web UI
- Tenant architectures
- User provisioning
- Authentication
- Password policy
- Enabling and disabling the password policy
- Configuring change password behavior
- Implementing an account lockout policy
- Transactions
- Implementing IP address whitelists
- Configuring row limit throttling
- Configuring throttling for OData queries
- Configuring CORS behavior
- FIPS (Federal Information Processing Standard)
- Data source logging
- Using third party JDBC drivers with Hybrid Data Pipeline
- Configuring Hybrid Data Pipeline to authorize client applications using OAuth 2.0
- Integrating Hybrid Data Pipeline with a Google OAuth 2.0 authorization flow to access Google Analytics
- Troubleshooting

Initial login with default user accounts

You must specify passwords for the default d2cadmin and d2cuser accounts during installation of the Hybrid Data Pipeline server. Best practices recommend that you follow the Hybrid Data Pipeline default password policy when specifying these account passwords. When initially logging in to the Web UI or using Hybrid Data Pipeline APIs, you must authenticate as one of these users.

The d2cadmin account has the default System Administrator role. The System Administrator role has all Hybrid Data Pipeline permissions. The d2cuser account has the default User role. The User role has a set of permissions associated with standard user tasks. (See Permissions and default roles on page 59 for details.) These default roles cannot be deleted. However, the users associated with them can be modified through the Web UI or Hybrid Data Pipeline APIs.

As a matter of best practices, you should consider removing the default d2cadmin and d2cuser accounts. To remove the default d2cadmin account, you must create at least one other user with the Administrator permission. When you log in through the new account that has the Administrator permission, you can then remove the default d2cadmin account. Hybrid Data Pipeline requires that at least one user have the Administrator permission. However, as a matter of best practices, more than one user should have Administrator permission at any time. For more information on provisioning users, see Tenant architectures on page 82 and User provisioning on page 107.
Permissions and default roles

Hybrid Data Pipeline user accounts are required to have at least one role. A user account with a given role inherits all the permissions associated with that role. Roles can be assigned and managed either through the Users API or the Web UI. However, the users API must be used to associate a permission directly with a user account. The permissions for a user account are the sum of the permissions granted to the role(s) associated with the account and the permissions granted explicitly on the account.

Hybrid Data Pipeline provides three default roles in the system tenant: System Administrator, Tenant Administrator, and User. As detailed in the table below, the System Administrator role has all permissions, the Tenant Administrator role has tenant and user permissions, and the User role has only user permissions. These roles cannot be deleted, and only the users associated with them can be modified.

When building out a Hybrid Data Pipeline environment, it can be useful for administrators to consider permissions in terms of the following categories.

- **User** permissions support the ability of users to create and manage their own data sources directly through the Web UI, the Management API, or both.

- **Tenant** permissions support the ability of administrators to provision and manage users. The OnBehalfOf permission allows administrators to create and manage resources on behalf of users. This on-behalf-of functionality allows administrators to obscure or conceal the service from users.

- **Elevated** permissions support the ability of administrators to use administrative features, such as throttling and logging. The operations associated with these permissions can affect all users of the system and may not be isolated on a tenant-by-tenant basis.

---

**Important:** To administer user accounts and other resources that belong to a tenant, a tenant administrator must be given explicit administrative access to the given tenant. In the Web UI, administrative access to a tenant can be granted by editing a user account via the Manage Users view on page 65. With the API, administrative access can be granted either by updating the tenants administered for a user via the Users API or by updating the list of administrators for a tenant via the Tenant API.

---

**Note:** A subset of permissions can be set on data sources. See Data source permissions on page 1321 for details.

<table>
<thead>
<tr>
<th>Permission</th>
<th>System admin</th>
<th>Tenant admin</th>
<th>User</th>
<th>Category</th>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateDataSource</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>1</td>
<td>May create new data sources</td>
</tr>
<tr>
<td>ViewDataSource</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>2</td>
<td>May view the details of any data source they own</td>
</tr>
<tr>
<td>Permission</td>
<td>System admin</td>
<td>Tenant admin</td>
<td>User</td>
<td>Category</td>
<td>ID</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------</td>
<td>--------------</td>
<td>-------</td>
<td>----------</td>
<td>----</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ModifyDataSource</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>3</td>
<td>May modify or update any data source they own</td>
</tr>
<tr>
<td>DeleteDataSource</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>4</td>
<td>May delete any data source they own</td>
</tr>
<tr>
<td>UseDataSourceWithJDBC</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>5</td>
<td>May connect to any data source they own with the JDBC driver</td>
</tr>
<tr>
<td>UseDataSourceWithODBC</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>6</td>
<td>May connect to any data source they own with the ODBC driver</td>
</tr>
<tr>
<td>UseDataSourceWithOData</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>7</td>
<td>May make OData requests to any data source they own</td>
</tr>
<tr>
<td>WebUI</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>8</td>
<td>May use the Web UI with data sources they own. Operations on the data source through the Web UI will be limited based on the permissions they have been granted</td>
</tr>
<tr>
<td>ChangePassword</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>9</td>
<td>May use the Web UI to change their password</td>
</tr>
<tr>
<td>SQLEditorWebUI</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>10</td>
<td>May query the data sources they own with the SQL Editor in the Web UI</td>
</tr>
<tr>
<td>MgmtAPI</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>user</td>
<td>11</td>
<td>May use the Management API</td>
</tr>
<tr>
<td>CreateUsers</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>13</td>
<td>May create users in administered tenants</td>
</tr>
<tr>
<td>ViewUsers</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>14</td>
<td>May get lists of users and their information in administered tenants</td>
</tr>
<tr>
<td>ModifyUsers</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>15</td>
<td>May modify user information in administered tenants</td>
</tr>
<tr>
<td>DeleteUsers</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>16</td>
<td>May delete users in administered tenants</td>
</tr>
<tr>
<td>CreateRole</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>17</td>
<td>May create roles in administered tenants</td>
</tr>
<tr>
<td>ViewRole</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>18</td>
<td>May get lists of roles and their information in administered tenants</td>
</tr>
<tr>
<td>ModifyRole</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>19</td>
<td>May modify role information in administered tenants</td>
</tr>
<tr>
<td>DeleteRole</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>20</td>
<td>May delete roles in administered tenants</td>
</tr>
<tr>
<td>OnBehalfOf</td>
<td>x</td>
<td>x</td>
<td></td>
<td>tenant</td>
<td>21</td>
<td>May use ?user=&lt;user&gt; to manage user's data sources in administered tenants</td>
</tr>
<tr>
<td>Permission</td>
<td>System admin</td>
<td>Tenant admin</td>
<td>User</td>
<td>Category</td>
<td>ID</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------</td>
<td>--------------</td>
<td>------------</td>
<td>-----------</td>
<td>----</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Configurations</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>22</td>
<td></td>
<td>May view and modify system configuration values</td>
</tr>
<tr>
<td>CORS whitelist</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>23</td>
<td></td>
<td>May view and modify the CORS whitelist</td>
</tr>
<tr>
<td>Logging</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>24</td>
<td></td>
<td>May view and modify logging settings</td>
</tr>
<tr>
<td>TenantAPI</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>25</td>
<td></td>
<td>May use the Tenant API to create, view, modify or delete tenants</td>
</tr>
<tr>
<td>RegisterExternalAuthService</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>26</td>
<td></td>
<td>May create, view, modify, or delete authentication services in administered tenants</td>
</tr>
<tr>
<td>Limits</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>27</td>
<td></td>
<td>May see and modify limit values for administered tenants, users in administered tenants, and data sources of users in administered tenants</td>
</tr>
<tr>
<td>OAuth</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>28</td>
<td></td>
<td>May specify and update OAuth information that a data source uses for authentication</td>
</tr>
<tr>
<td>IPWhiteList</td>
<td>x</td>
<td></td>
<td>elevated</td>
<td>29</td>
<td></td>
<td>May create, view, modify or delete IP whitelists</td>
</tr>
<tr>
<td>Administrator</td>
<td>x</td>
<td></td>
<td>system admin</td>
<td>12</td>
<td></td>
<td>May use the Administrator API. A user with the Administrator permission has all permissions and access privileges across the system. This permission can only be granted to a user in the system tenant.</td>
</tr>
</tbody>
</table>

See also
- Tenant architectures on page 82
- User provisioning on page 107

Logging in to the Web UI

Logging in to the Web UI is a two step process. First, you must enter the URL of your Hybrid Data Pipeline instance in the address field of a supported browser. Then, you must enter your username and password at the Hybrid Data Pipeline login screen.

A URL includes the Web protocol, a server name, and a port number. For example:

https://MyServer:8443/hdpiui

The syntax for this URL can be described as follows.
webprotocol://servername:portnumber

where

webprotocol

is the Web protocol, such as HTTP or HTTPS, used to connect to your Hybrid Data Pipeline instance.

servername

is the name of the machine hosting the Hybrid Data Pipeline service, or the name of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service.

portnumber

is the port number of the machine hosting the Hybrid Data Pipeline service, or the port number of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service. For a standalone installation, the port number is specified as the Server Access Port during installation. For a load balancer installation, the port number must be either 80 for http or 443 for https. Whenever port 80 or 433 are used, it is not necessary to include the port number in the URL.

See also
Initial login with default user accounts on page 58
Using the Web UI on page 63
Using Hybrid Data Pipeline APIs on page 62

Using Hybrid Data Pipeline APIs

Hybrid Data Pipeline provides a representational state transfer (REST) application programming interface (API) for managing Hybrid Data Pipeline connectivity service resources.

Hybrid Data Pipeline APIs use HTTP Basic Authentication to authenticate user accounts. The Hybrid Data Pipeline user ID and password are encoded in the Authorization header. The Hybrid Data Pipeline user specified in the Authorization header is the authenticated user.

To execute REST calls, you must pass a valid REST URL and pass a valid username and password to authenticate with basic authentication. A REST URL must include a base and resource-specific information. The base includes the Web protocol, a server name, and a port number, while resource-specific information provides a path to a particular resource necessary for performing an API operation. For example:

https://MyServer:8443/api/mgmt/datasources

**Note:** The port number is only required if the Hybrid Data Pipeline server or load balancer is configured to use a port other than 443 for SSL or 80 for non-SSL connections.

The syntax for a REST URL can be described as follows.

webprotocol://servername:portnumber/resourceinfo

where

webprotocol

is the Web protocol, such as HTTP or HTTPS, used to connect to your Hybrid Data Pipeline instance.
servername

is the name of the machine hosting the Hybrid Data Pipeline service, or the name of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service.

portnumber

is the port number of the machine hosting the Hybrid Data Pipeline service, or the port number of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service. For a standalone installation, the port number is specified as the Server Access Port during installation. For a load balancer installation, the port number must be either 80 for http or 443 for https. Whenever port 80 or 433 are used, it is not necessary to include the port number in the URL.

resourceinfo

is resource-specific information that provides a path to a particular Hybrid Data Pipeline resource necessary to perform an API operation.

See also
Hybrid Data Pipeline API reference on page 1037
Initial login with default user accounts on page 58
User provisioning on page 107
Logging in to the Web UI on page 61

Using the Web UI

The Hybrid Data Pipeline Web UI consists of views which can be selected from the navigation bar to the left. Access to these views, and the ability to execute operations they support, depend on permissions granted to the user (see Permissions and default roles on page 59 for details). These views include:

• Manage Tenants
• Manage Users
• Manage Roles
• Data Sources
• SQL Editor
• Manage Limits
• Manage External Authentication
• System Configurations

See the following topics for details on these views and other features of the Web UI.

• Manage Tenants view on page 64
• Manage Users view on page 65
• Manage Roles view on page 67
• Data Sources view on page 69
• SQL Editor view on page 74
• Manage Limits view on page 76
• Manage External Authentication view on page 78
• System Configurations view on page 79
• User profile on page 81
• Changing your password in the Web UI on page 81
• Product information on page 81

Manage Tenants view

The Manage Tenants view provides a list of tenants with description and status information for each tenant. With the appropriate permissions, you can add, modify, and delete tenants using this view.

The Manage Tenants view is available to users with either set of the following permissions.

• Administrator (12) permission
• WebUI (8) and TenantAPI (25) permissions, and administrative access on tenants the user administers

The following table provides permissions and descriptions for each action in the Manage Tenants view.

---

**Note:** Any user with Administrator (12) permission may perform all actions.
<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create new tenant</td>
<td>Web UI (8) TenantAPI (25)</td>
<td>To create a new tenant, click + New Tenant. Define the tenant with settings under each of the following tabs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>General</strong> tab. Enter values in the given fields. The tenant name is required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Roles</strong> tab. Import roles from the parent tenant, if desired.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• <strong>Limits</strong> tab. Set throttling limits, if desired.</td>
</tr>
<tr>
<td>Edit a tenant</td>
<td>Administrative access for the tenant Web UI (8) TenantAPI (25)</td>
<td>To edit a tenant, select a tenant from the list of tenants. Then, select Edit from the Actions dropdown. Edit the tenant settings as desired.</td>
</tr>
<tr>
<td>Delete a tenant</td>
<td>Administrative access for the tenant Web UI (8) TenantAPI (25)</td>
<td>To delete a tenant, select the tenant you want to delete. Then, select Delete from the Actions dropdown. Confirm or cancel the delete operation in the dialog.</td>
</tr>
<tr>
<td>View tenant users</td>
<td>Administrative access for the tenant Web UI (8) ViewUsers (14) TenantAPI (25)</td>
<td>To view the users of a tenant, select the tenant from the list of tenants. Then, select View Users from the Actions dropdown. You are directed to the Manage Users view where a list of users belonging to the tenant is displayed. See Manage Users view on page 65 for details.</td>
</tr>
<tr>
<td>Transfer tenant users</td>
<td>Administrative access for the system tenant and the tenant to which user(s) will be transferred Web UI (8) ViewUsers (14) ModifyUsers (15) TenantAPI (25)</td>
<td>To transfer users from the system tenant to a child tenant, select the child tenant from the list of tenants. Then, select Transfer Users from the Actions dropdown. You are directed to the Transfer User From System Tenant page. Select each user you want to transfer to the child tenant, and choose a role for each user from the role dropdown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> Users can only be transferred from the system tenant to a child tenant.</td>
</tr>
</tbody>
</table>

### Manage Users view

The Manage Users view provides a list of users with roles and status information for a given tenant. With the appropriate permissions, you can add, update, and delete users using this view.

The Manage Users view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, ViewUsers (14) permission, ViewRole (18) permission, and administrative access on the tenant to which the users belong
The following table provides permissions and descriptions for each action in the Manage Users view.

**Note:** Any user with Administrator (12) permission may perform all actions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter users by tenant</td>
<td>Administrative access to multiple tenants</td>
<td>An administrator with administrative access to multiple tenants will have the option of selecting the tenant for which he or she wants to view or manage users. Select the tenant for which you want to view users from the <strong>Select Tenant</strong> dropdown.</td>
</tr>
<tr>
<td>Create a new user</td>
<td>Administrative access for the tenant</td>
<td>To create a new user, click <strong>+ New User</strong>. Define the user with settings under each of the following tabs.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td>• <strong>General</strong> tab. Enter values in the given fields. User name and role are required.</td>
</tr>
<tr>
<td></td>
<td>CreateUsers (13)</td>
<td>• <strong>Authentication Setup</strong> tab. The required information depends on the type of authentication you are using. See <strong>Authentication</strong> on page 143 for details.</td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td>• <strong>Limits</strong> tab. Set throttling limits, if desired.</td>
</tr>
<tr>
<td></td>
<td>ViewRole (18)</td>
<td>• <strong>Tenant Admin Access</strong> tab. Grant the user administrative access to tenant(s), if desired.</td>
</tr>
</tbody>
</table>
### Manage Roles view

The **Manage Roles** view provides a list of roles for a given tenant. With the appropriate permissions, you can add, update, and delete roles using this view.

The **Manage Roles** view is available to users with either set of the following permissions:

- Administrator (12) permission
- WebUI (8) permission, ViewRole (18) permission, and administrative access on the tenant to which the role(s) belong
The following table provides permissions and descriptions for each action in the Manage Roles view.

**Note:** Any user with Administrator (12) permission may perform all actions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter roles by tenant</td>
<td>Administrative access to multiple tenants</td>
<td>An administrator with administrative access to multiple tenants will have the option of selecting the tenant for which he or she wants to view or manage roles. Select the tenant for which you want to view roles from the Select Tenant dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewRole (18)</td>
<td></td>
</tr>
<tr>
<td>Create a new role</td>
<td>Administrative access for the tenant</td>
<td>To create a new role, click + New Role. Provide a name and description for the new role. Then, select permissions to define the role.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CreateRole (17)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewRole (18)</td>
<td></td>
</tr>
</tbody>
</table>
### Data Sources view

The Data Sources view allows you to manage data sources and data source groups. The Data Sources view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) and ViewDataSource (2) permissions

The Data Sources view consists of the following pages.

- Data Sources
- Data Source Groups

#### Data Sources

The Data Sources page enables you to create, edit, and delete data source definitions. A data source definition configures the connection between Hybrid Data Pipeline and a data store.
The following table provides permissions and descriptions for basic actions in the Data Sources page. For detailed information on creating data sources, see Creating data sources with the Web UI on page 224 and How to create a data source in the Web UI on page 225.

**Note:** With the appropriate permissions, administrators can view data sources owned by other users through the Web UI. However, administrators cannot create, modify, or delete data sources owned by other users through the Web UI. To create, modify, or delete data sources that belong to other users, administrators must use Hybrid Data Pipeline APIs. See Data Sources API on page 1277 and Managing resources on behalf of users on page 1281 for further details.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
</table>
| Filter data sources by tenant | Administrative access to multiple tenants  
Web UI (8)  
ViewDataSource (2)  
ViewUsers (14) | An administrator with administrative access to multiple tenants will have the option of filtering by tenants to view data sources owned by a given user. Select the tenant in which the user resides from the Select Tenant dropdown.  
**Note:** Any user with the Administrator (12) permission can view the data sources of any user across all tenants. |
<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter data sources by user</td>
<td>Administrative access to the tenant</td>
<td>To filter data sources by user, select the user whose data sources you want to view from the Select User dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Any user with the Administrator (12) permission can view the data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sources of any user across all tenants.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search for a data source</td>
<td>Web UI (8)</td>
<td>Use the search field in the upper right to filter data sources by name, data store, and description.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td>Create a new data source</td>
<td>Web UI (8)</td>
<td>To create a new data source, click + New Data Source. See How to create a data source in the Web UI on page 225 for details.</td>
</tr>
<tr>
<td></td>
<td>CreateDataSource (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td>Modify a data source</td>
<td>Web UI (8)</td>
<td>To modify a data source, select the data source from the list of data sources. Then, select Edit from the Actions dropdown. Edit the data source as desired.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ModifyDataSource (3)</td>
<td></td>
</tr>
<tr>
<td>Delete a data source</td>
<td>Web UI (8)</td>
<td>To delete a data source, select the data source you want to delete. Then, select Delete from the Actions dropdown. Confirm or cancel the delete operation in the dialog.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DeleteDataSource (4)</td>
<td></td>
</tr>
<tr>
<td>Test a data source</td>
<td>Web UI (8)</td>
<td>To run queries against a data source through the Web UI, select the data source. Then, select SQL Testing from the Actions dropdown. You are directed to the SQL Editor view where you review schema and execute a SQL statement against the data source.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sqleWebUI (10)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>At least one of the following query permissions:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• UseDataSourceWithJDBC (5)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• UseDataSourceWithODBC (6)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• UseDataSourceWithOData (7)</td>
<td></td>
</tr>
</tbody>
</table>

Using the Web UI
### Data Source Groups

The **Data Source Groups** page enables you to combine OData enabled data sources into a single data source group. You can create, edit, and delete data source groups from this page.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sync OData Schema</td>
<td>Web UI (8) ViewDataSource (2)</td>
<td>OData enabled data sources maintain an OData model. The OData model should be refreshed whenever the schema of the data source schema has been changed. To refresh the OData model, click the sync icon ⚙️. For details, see <a href="#">Configuring data sources for OData connectivity and working with data source groups</a> on page 622.</td>
</tr>
<tr>
<td>Obtain OData URI</td>
<td>Web UI (8) ViewDataSource (2)</td>
<td>To obtain the OData URI for an OData enabled data source, copy the link associated with the link icon ⚙️.</td>
</tr>
<tr>
<td>Configure data source logging</td>
<td>Web UI (8) ViewDataSource (2) Logging (24)</td>
<td>To configure data source logging, click the settings icon ⚒️. You are directed to the <strong>Logging Settings</strong> page. Set logging and privacy levels as desired.</td>
</tr>
</tbody>
</table>

The following table provides permissions and descriptions for basic actions in the **Data Source Groups** page. For detailed information on creating OData enabled data sources and data source groups, see [Configuring data sources for OData connectivity and working with data source groups](#) on page 622.
<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Filter data source groups by tenant</strong></td>
<td>Administrative access to multiple tenants</td>
<td>An administrator with administrative access to multiple tenants will have the option of filtering by tenants to view data source groups owned by a given user. Select the tenant in which the user resides from the Select Tenant dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td>Note: Any user with the Administrator (12) permission can view the data source groups of any user across all tenants.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Filter data source groups by user</strong></td>
<td>Administrative access to the tenant</td>
<td>To filter data source groups by user, select the user whose data source groups you want to view from the Select User dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td>Note: Any user with the Administrator (12) permission can view the data source groups of any user across all tenants.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Search for a data source group</strong></td>
<td>Web UI (8)</td>
<td>Use the search field in the upper right to filter data source groups by name, data store, and description.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td><strong>Create a new data source group</strong></td>
<td>Web UI (8)</td>
<td>To create a new data source group, click + New Group. See Creating a data source group on page 635 for details.</td>
</tr>
<tr>
<td></td>
<td>CreateDataSource (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td><strong>Modify a data source group</strong></td>
<td>Web UI (8)</td>
<td>To modify a data source group, select the group. Then, select Edit from the Actions dropdown. Edit the group as desired.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ModifyDataSource (3)</td>
<td></td>
</tr>
</tbody>
</table>
### SQL Editor view

The SQL Editor view allows users to browse schemas and to query data associated with a data source.

The SQL Editor view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, ViewDataSource (2) permission, SQLEditorWebUI (10) permission, and, to query data sources, at least one of the following query permissions:
  - UseDataSourceWithJDBC (5)
  - UseDataSourceWithODBC (6)
  - UseDataSourceWithOData (7)

---

3 For backend data stores that support schemas, the Metadata Exposed Schemas option can be used to restrict the exposed schemas to a single schema. Metadata Exposed Schemas only affects the metadata that is displayed in the Schema navigation pane. SQL queries can still be executed against tables in other schemas. For details, see the parameters topic for your data source type.
The following table provides permissions and descriptions for actions in the **SQL Editor** view. To perform any action from this view, begin by selecting a data source from the **Select a Data Source** dropdown.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explore the schema and tables associated with the data source</strong></td>
<td>WebUI (8) ViewDataSource (2) SQLEditorWebUI (10)</td>
<td>To begin, a data source must be selected from the <strong>Select a Data Source</strong> dropdown. To view schema tables, click the a schema carrot in the <strong>Schema Tree</strong> panel. Click on a table to view the details of a table in the <strong>Table Details</strong> panel. Views and procedures that reside in the schema may also be listed.</td>
</tr>
<tr>
<td><strong>Execute a SQL statement against the data source</strong></td>
<td>Web UI (8) ViewDataSource (2) SQLEditorWebUI (10)</td>
<td>To begin, a data source must be selected from the <strong>Select a Data Source</strong> dropdown. To run a query against the data source, enter the SQL statement in the field provided in the <strong>Editor</strong> panel. Then click <strong>Execute</strong> to run the query. SQL query results will be returned in the <strong>Results</strong> panel. Note: Queries made via the <strong>SQL Editor</strong> view time out after 6 minutes. Therefore, to validate a data source connection, you should execute queries that require less processing time. For large queries, only the first 200 results are shown.</td>
</tr>
</tbody>
</table>
Manage Limits view

The Manage Limits view allows you to view and set limits for features such as throttling and logging.

In the Manage Limits view, limits can be set at either the system or tenant level. System limits apply to behavior across Hybrid Data Pipeline and override default behavior, while tenant limits apply to the resources of a given tenant and override default behavior and system limits. Most limits can only be configured at the system level. However, some limits, such as MaxFetchRows and MaxConcurrentQueries, can be configured at any level.

**Note:** Limits can also be specified for users and data sources. User limits can be set either through the Manage Users view on page 65 or the Limits API on page 1071. User limits override default, system, and tenant limits. Data source limits can only be set via the Limits API on page 1071. Data source limits override all other limits.

The Manage Limits view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, Limits (27) permission, and administrative access on the given tenant

The table below provides descriptions for limits that may be set via the Manage Limits view.

**Note:**

- Throttling limits can be set either for the system tenant or any child tenant across the system.
- Log Management, Data Usage Meter, and Security limits can only be set for the system.
- To set system limits, the system tenant must be selected from the Tenant dropdown. The user must have the Administrator (12) permission.
• To set tenant limits, the child tenant must be selected from the Tenant dropdown. The user must have either the Administrator (12) permission, or WebUI (8), Limits (27) permissions, and administrative access on the given tenant.

• Tenant limits can also be set via the Manage Tenants view on page 64.

<table>
<thead>
<tr>
<th>Category</th>
<th>Limit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throttling</td>
<td>MaxFetchRows</td>
<td>Maximum number of rows allowed to be fetched for a single query.</td>
</tr>
<tr>
<td>Throttling</td>
<td>ODataMaxConcurrentQueries</td>
<td>Maximum number of concurrent active OData queries per data source.</td>
</tr>
<tr>
<td>Throttling</td>
<td>TransactionTimeout</td>
<td>The number of seconds the system allows a transaction to be idle before rolling it back.</td>
</tr>
<tr>
<td>Throttling</td>
<td>XdbcMaxResponse</td>
<td>Approximate maximum size of JDBC/ODBC HTTP result data in KB.</td>
</tr>
<tr>
<td>Log Management</td>
<td>LogRetentionDays</td>
<td>Number of days log files should be retained.</td>
</tr>
<tr>
<td>Log Management</td>
<td>MonitorRetentionDays</td>
<td>Number of days monitor details should be retained</td>
</tr>
<tr>
<td>Data Usage Meter</td>
<td>UserMeterRetentionDays</td>
<td>Number of days user meter details should be retained</td>
</tr>
<tr>
<td>Data Usage Meter</td>
<td>UserMeterWriteInterval</td>
<td>The number of seconds the system waits before scanning sessions for current metrics. A lower setting will result in more rows written to the meter table</td>
</tr>
<tr>
<td>Data Usage Meter</td>
<td>UserMeterMaxAge</td>
<td>The number seconds the system waits before writing out meter records. A lower setting will result in the rows written to meter table to occur more frequently</td>
</tr>
<tr>
<td>Security</td>
<td>PasswordLockoutInterval</td>
<td>The duration, in seconds, for counting the number of consecutive failed authentication attempts.</td>
</tr>
<tr>
<td>Security</td>
<td>PasswordLockoutLimit</td>
<td>The number of consecutive failed authentication attempts that are allowed before locking the user account.</td>
</tr>
<tr>
<td>Security</td>
<td>PasswordLockoutPeriod</td>
<td>The duration, in seconds, for which a user account will not be allowed to authenticate to the system when the PasswordLockoutLimit is reached.</td>
</tr>
<tr>
<td>Security</td>
<td>OAuthAccessTokenDuration</td>
<td>The duration, in minutes, for which a Access token is valid.</td>
</tr>
<tr>
<td>Category</td>
<td>Limit</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Security</td>
<td>OAuthAccessTokenCacheSize</td>
<td>Number of OAuth access tokens to be cached in memory for OAuth Authentication. By default up to 2000 tokens will be cached in memory.</td>
</tr>
<tr>
<td>Security</td>
<td>CORSBehavior</td>
<td>Configuration parameter for CORS behavior. Setting the value to 0 disables the CORS filter. Setting the value to 1 enables the CORS filter. Setting the value to 2 enables the CORS filter with the whitelist option.</td>
</tr>
</tbody>
</table>

Manage External Authentication view

The Manage External Authentication view allows you to add, update, and delete an external authentication service. The external authentication service must first be implemented by a system administrator as described in Authentication on page 143. Once the service has been implemented, it can be added to a tenant.

The Manage External Authentication view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, RegisterExternalAuthService (26) permission, and administrative access on the given tenant

The following table provides permissions and descriptions for actions in the Manage External Authentication view.
Note: Any user with Administrator (12) permission may perform all actions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter roles by tenant</td>
<td>Administrative access to multiple tenants</td>
<td>An administrator with administrative access to multiple tenants will have the option of selecting the tenant for which he or she wants to view or manage external authentication services. Select the tenant for which you want to view authentication services from the Select Tenant dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RegisterExternalAuthService (26)</td>
<td></td>
</tr>
<tr>
<td>Register an external authentication service</td>
<td>Administrative access for the tenant</td>
<td>To register an authentication service with the tenant, click + New Service. Provide the following information, and then click Save.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td>• The name and description of the service</td>
</tr>
<tr>
<td></td>
<td>RegisterExternalAuthService (26)</td>
<td>• The service type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For Java plugin service provide:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• The class name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Attributes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• For LDAP service provide:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Target URL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Service Authentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Security Principal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Other Attributes</td>
</tr>
<tr>
<td>Edit an external authentication service</td>
<td>Administrative access for the tenant</td>
<td>To edit an authentication service, select the service. Then, select Edit from the Actions dropdown. Edit the service as desired.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RegisterExternalAuthService (26)</td>
<td></td>
</tr>
<tr>
<td>Delete an external authentication service</td>
<td>Administrative access for the tenant</td>
<td>To delete a service, select the service you want to delete. Then, select Delete from the Actions dropdown. Confirm or cancel the delete operation in the dialog.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RegisterExternalAuthService (26)</td>
<td></td>
</tr>
</tbody>
</table>

System Configurations view

The System Configurations view can be used to set a number of configurations across the Hybrid Data Pipeline system. This view is only available to users with the Administrator (12) permission (system administrators).
The following table provides descriptions of the options available via the **System Configurations** view.

<table>
<thead>
<tr>
<th>Option</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delimiter</td>
<td>Administrator (12)</td>
<td>Specifies a delimiter to be used between the user name and authentication service name. In the following example, the</td>
</tr>
<tr>
<td>Secure Password Change</td>
<td>Administrator (12)</td>
<td>Specifies whether the current password is required in order to update the password of the logged-in user. The default value is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Default OData Version</td>
<td>Administrator (12)</td>
<td>Sets the default OData version for new data sources.</td>
</tr>
<tr>
<td>Default Entity Name</td>
<td>Administrator (12)</td>
<td>Sets the default entity name mode for OData V4 data sources. For details, see <a href="#">Configuring data sources for OData connectivity and working with data source groups</a> on page 622.</td>
</tr>
<tr>
<td>JDBC DataStore</td>
<td>Administrator (12)</td>
<td>Enables the third party JDBC data store feature. The default value is <strong>ON</strong>. For details, see <a href="#">Using third party JDBC drivers with Hybrid Data Pipeline</a> on page 182.</td>
</tr>
<tr>
<td>Password Policy</td>
<td>Administrator (12)</td>
<td>Enables the default password policy. The default value is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Option</td>
<td>Permissions</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>System Monitor Details</td>
<td>Administrator (12)</td>
<td>Determines how the system persists monitor details.</td>
</tr>
<tr>
<td>IP WhiteList Filtering</td>
<td>Administrator (12)</td>
<td>Enables the whitelist filtering feature. The default value is ON. See Implementing IP address whitelists on page 164 for details.</td>
</tr>
</tbody>
</table>

**Product information**

Users can access product information by clicking the question mark icon and selecting **About**.

![About Hybrid Data Pipeline](image)

The **About Hybrid Data Pipeline** window displays installation and version information.

![About Hybrid Data Pipeline](image)

**User profile**

The down arrow next to the username in the upper right hand corner of the Web UI opens a dropdown menu. Users can change their passwords by selecting the **Change Password** item, or log out by selecting the **Log Out** item.

**Changing your password in the Web UI**

Take the following steps to change your password in the WebUI.
**Tenant architectures**

A Hybrid Data Pipeline system administrator can develop either a single-tenant or multitenant architecture. In a single-tenant architecture, the system administrator creates user accounts in the default *system tenant*. In a multitenant architecture, the system administrator first creates one or more child tenants in the default system tenant. Then, the system administrator can create user accounts in either the system tenant or any one of the child tenants. The user accounts that reside in one tenant are isolated from those in other tenants.

When establishing a tenant architecture, the system administrator should consider the roles users and other administrators will assume in the Hybrid Data Pipeline environment. As detailed in Permissions and default roles on page 59, Hybrid Data Pipeline provides three default roles: System Administrator, Tenant Administrator, and User. These roles can be used in either a single-tenant or multitenant architecture. In the context of these roles, the system administrator has full permissions and administrative access across the system, while the tenant administrator can assume responsibility for provisioning and managing user accounts in tenants for which he or she has administrative access.

**Important:** To administer user accounts and other resources that belong to a tenant, a tenant administrator must be given explicit administrative access to the given tenant. In the Web UI, administrative access to a tenant can be granted by editing a user account via the Manage Users view on page 65. With the API, administrative access can be granted either by updating the tenants administered for a user via the Users API or by updating the list of administrators for a tenant via the Tenant API.

The following topics describe single-tenant and multitenant architectures in greater detail, including how administrative roles can be applied in each.

- Single-tenant environment
- Multitenant environment
**Single-tenant environment**

Tenancy is mostly transparent in a single-tenant environment where users and features are managed from the default system tenant. Nevertheless, tenant and elevated permissions were introduced with support for multitenancy. Tenant permissions support the ability of administrators to provision and manage users, while elevated permissions support the ability of administrators to execute other administrative tasks, such as throttling and logging. By granting such permissions to other users, the system administrator can delegate administrative tasks and responsibilities.

**Note:** As an alternative to using the default system tenant, a system administrator could create a child tenant in the system tenant. This child tenant could function as a single, dedicated tenant from which users and features are managed.

See the following topics on how to set up a single-tenant environment.

- Using the Web UI to set up a single-tenant environment on page 83
- Using the APIs to set up a single-tenant environment on page 84

**See also**
- User provisioning on page 107
- Permissions and default roles on page 59

**Using the Web UI to set up a single-tenant environment**

The following steps show how you can set up a single-tenant environment using the Web UI.

**Note:** It is assumed that users and features will be managed from the default system tenant. Therefore, there is no step to create a child tenant.

1. Create administrator roles.
   a) Navigate to the Manage Roles view by clicking the manage roles icon.
   b) Click + New Role.
   c) Provide a name and description for the role.
   d) Select permissions to define the role.
   e) Click Save.

2. Create administrator users.
   a) Navigate to the Manage Users view by clicking the manage users icon.
   b) Click + New User.
   c) Define the user with settings under each of the following tabs.
      - Under the General tab, enter a user name and assign the role you have created for the user.
      - Under the Authentication Setup tab, configure authentication settings.
      - Under the Limits tab, configure limits as desired. Note that user limits override system limits.
• Under the Tenant Admin Access tab, grant the user administrative access to the system tenant.

d) Click Save.

3. Set system configurations.

   a) Navigate to the System Configurations view by clicking the system configurations icon.
   b) Configure options as desired. See System Configurations view on page 79 for option descriptions.
   c) Click Save.

4. Set limits.

   a) Navigate to the Manage Limits view by clicking the manage limits icon.
   b) Set limits as desired. See Manage Limits view on page 76 for limit descriptions.
   c) Click Save.

Using the APIs to set up a single-tenant environment

The following operations show how you can set up a single-tenant environment using Hybrid Data Pipeline APIs.

Note: It is assumed that users and features will be managed from the default system tenant. Therefore, there is no step to create a child tenant.

• Retrieving valid roles in the system tenant
• Create a user with the Tenant Administrator role
• Grant the administrator user administrative access to the system tenant
• Create a new role with tenant and elevated permissions
• Assign the new role to the administrator user
• Retrieving and setting system configurations
• Retrieving and setting limits

Retrieving valid roles in the system tenant

The following GET operation retrieves the valid roles and their IDs for the system tenant in a single-tenant environment. Role IDs can then be used to assign roles to users.

Request

GET https://MyServer:8443/api/admin/roles

Response Payload

```json
{  
"roles": [  
  
  {  
"id": 1,  
"name": "System Administrator",  
"tenantId": 1,  
"description": "This role has all permissions. This role cannot be
```
modified or deleted.
}

{
  "id": 2,
  "name": "User",
  "tenantId": 1,
  "description": "This role has the default permissions that a normal
  user will be expected to have."
},

{
  "id": 3,
  "name": "Tenant Administrator",
  "tenantId": 1,
  "description": "This role has all the tenant administrator
  permissions."
}

Create a user with the Tenant Administrator role

The ID for the Tenant Administrator role (3) can then be used to create a user with the Tenant Administrator
role, as shown in the following POST operation. The user inherits the permissions associated with this role.

Request

POST https://MyServer:8443/api/admin/users

Request Payload

{
  "userName": "TenantAdmin",
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "password": "<password>",
    "passwordStatus": 1,
    "passwordExpiration": "2020-01-01 00:00:00"
  },
  "permissions": {
    "roles": [3]
  }
}

Response Payload

{
  "id": 87,
  "userName": "TenantAdmin",
  "tenantId": 1,
  "tenantName": "Root",
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "passwordStatus": 1,
    "passwordExpiration": "2020-01-01 00:00:00.0"
  },
  "permissions": {
    "roles": [3]
  }
}
Grant the administrator user administrative access to the system tenant

In addition to being granted the Tenant Administrator role, the tenant administrator must be granted administrative access to the system tenant. The following Users API request grants user account 87 administrative access to the system tenant.

**Note:** Administrative access to the system tenant can also be managed by updating the list of administrators via the Tenant API.

**Request**

PUT https://MyServer:8443/api/admin/users/87/tenantsadministered

Request Payload

```json
{
   "tenantsAdministered": [1]
}
```

Response Payload

```json
{
   "tenantsAdministered": [1]
}
```

Create a new role with tenant and elevated permissions

The following POST request creates the new Tenant Admin Plus role. The new role has all user and tenant permissions plus the Logging (24), Limits (27), and OAuth (28) permissions.

**Request**

POST https://MyServer:8443/api/admin/roles

Request Payload

```json
{
   "name": "Tenant Admin Plus",
   "description": "This role has all the tenant administrator permissions plus elevated permissions.",
   "permissions": [
      1,
      2,
      3,
      4,
      5,
      6,
      24,
      27,
      28
   ]
}
```
Assign the new role to the administrator user

The following PUT assigns the new Tenant Admin Plus role to the administrator user. The user inherits the permissions associated with this role. Note that the ID of the Tenant Admin Plus role (42) was provided in the response payload when the role was created. Also, note that any existing roles and permissions are removed by this operation.

Request

PUT https://MyServer:8443/api/admin/users/87/permissions
Request Payload

```json
{
   "roles": [42],
   "permissions": []
}
```

Response Payload

```json
{
   "roles": [42]
}
```

Retrieving and setting system configurations

The following GET operation retrieves a list of system configurations.

Request

GET https://MyServer:8443/api/admin/configurations

Response Payload

Note: See System Configurations API on page 1124 for a complete list of system configurations and their descriptions.

```json
{
   "configurations": [  
   {  
       "id": 1,  
       "description": "Delimiter between user name and authentication service/configuration name",  
       "value": null  
   },  
   {  
       "id": 2,  
       "description": "Enable Secure Password Change, when value is set to true, the change password api will require a valid old password in order to update the logged in user password.",  
       "value": "true"  
   },  
   ...  
   {  
       "id": 8,  
       "description": "Configure whitelist filtering. Enables filtering when value is set to 'true'. Default value is "true" ",  
       "value": "true"  
   }
   ]
}
```

The following PUT operation disables IP address whitelists. The number 8 is the ID of the IP address whitelist feature.

Request

PUT https://MyServer:8443/api/admin/configurations/8

Request Payload

```json
{
   "value": "false"
}
```
Retrieving and setting limits
The following GET operation retrieves a list of limits.

Request

GET https://MyServer:8443/api/admin/limits

Response Payload

Note: See Limits API on page 1071 for a complete list of limits and their descriptions.

```json
{
   "limits": [
      {
         "id": 1,
         "name": "MaxFetchRows",
         "description": "Maximum number of rows allowed to be fetched for a single query",
         "minValue": 1,
         "maxValue": 9000000000000000000,
         "defaultValue": 9000000000000000000,
         "validForLimits": 15
      },
      ...
      {
         "id": 6,
         "name": "ODataMaxConcurrentQueries",
         "description": "Maximum number of concurrent active queries per data source",
         "minValue": 0,
         "maxValue": 9000000000000000000,
         "defaultValue": 0,
         "validForLimits": 15
      },
      ...
   ]
}
```

The following POST creates a system-level limit of 50000 queries. The number 6 is the ID of the ODataMaxConcurrentQueries limit. The payload passes 50000 as the value for this limit.

Request

POST https://MyServer:8443/api/admin/limits/system/6

Request Payload

```json
{
   "value": 50000
}
```

See also
User provisioning on page 107
Users API on page 1146
Roles API on page 1112
System Configurations API on page 1124
Limits API on page 1071
Multitenant environment

Multitenancy allows system administrators to isolate groups of users, such as organizations or departments, hosted by the Hybrid Data Pipeline service. The system administrator maintains a physical instance of Hybrid Data Pipeline, while each tenant (group of users) is provided with its own logical instance of the service. To create a multitenant environment, the system administrator creates child tenants in the default system tenant. The system administrator can then proceed with setting up administrative and support structures for maintaining the Hybrid Data Pipeline environment. The administration of tenants follows two general patterns: system-level administration and tenant-level administration.

In system-level administration, a system administrator may want to delegate or share user provisioning and other administrative tasks with a tenant administrator who can manage user accounts and enable supported features across multiple tenants. In this instance, the system administrator creates tenant administrators in the system tenant with user management permissions and administrative access to the tenants they will manage. These tenant administrators are able to manage users, data sources, and other resources across multiple tenants.

In tenant-level administration, the system administrator delegates user provisioning and other administrative tasks to tenant administrators who belong to one of many tenants. For example, a Hybrid Data Pipeline provider may host several external organizations where it is appropriate for the organizations themselves to provision users and administer data access. In this scenario, the system administrator would create tenant administrators who reside in the tenants they administer, thus isolating administrative tasks such as user provisioning from one tenant to another. For tenant-level administration, tenant administrators must have administrative access to the tenants in which they reside, as well as user management and other permissions as needed.

Note that system-level and tenant-level administration are not mutually exclusive. For example, a system administrator might want to delegate and isolate the administration of tenants, but also provision support personnel to work with resources across multiple tenants.

The following topics provide information on creating multitenant environments.

- Setting up a multitenant environment with system-level administration on page 90
- Setting up a multitenant environment with tenant-level administration on page 98

Setting up a multitenant environment with system-level administration

A system administrator can take the following general steps to set up a multitenant environment with system-level administration.

1. Create child tenants.
2. Create administrator roles in the system tenant.
3. Create tenant administrators in the system tenant.
4. Set system configurations and limits.

See the following topics for details.

- Using the Web UI to set up a multitenant environment with system-level administration on page 90
- Using the APIs to set up a multitenant environment with system-level administration on page 92

Using the Web UI to set up a multitenant environment with system-level administration

Take the following steps to set up a multitenant environment with system-level administration using the Web UI.

1. Create tenants.
a) Navigate to the **Manage Tenants** view by clicking the manage tenants icon.
b) Click **+New Tenant**.
c) Under the **General** tab, enter a name and description for the tenant.
d) Under the **Roles** tab, select any roles that you created in the system tenant that you want to import to the new tenant.
e) Under the **Limits** tab, specify any limits that you want to set for the tenant. These limits will override limits at the system level.
f) Click **Save**.

2. Create administrator roles.

a) Navigate to the **Manage Roles** view by clicking the manage roles icon.
b) For **Tenant**, select **System** from the dropdown.
c) Click **+New Role**.
d) Provide a name and description for the role.
e) Select permissions to define the role.
f) Click **Save**.

3. Create administrator users.

a) Navigate to the **Manage Users** view by clicking the manage users icon.
b) For **Tenant**, select **System** from the dropdown.
c) Click **+New User**.
d) Define the user with settings under each of the following tabs.
   - Under the **General** tab, enter a user name and assign the role you have created for the user.
   - Under the **Authentication Setup** tab, configure authentication settings.
   - Under the **Limits** tab, configure limits as desired. Note that user limits override system limits.
   - Under the **Tenant Admin Access** tab, grant the user administrative access to any tenants they will be administering.
e) Click **Save**.

4. Set system configurations. These configurations apply to all tenants across the system.

a) Navigate to the **System Configurations** view by clicking the system configurations icon.
b) Configure options as desired. See **System Configurations view** on page 79 for option descriptions.
c) Click **Save**.

5. Set system limits.

a) Navigate to the **Manage Limits** view by clicking the manage limits icon.
b) For **Tenant**, select **System** from the dropdown.
c) Set limits as desired. Limits set on tenants will override limits set at the system level.

d) Click Save.

Results:
You have created child tenants in the system tenant. In addition, you have created tenant administrators who reside in the system tenant. These administrators can perform administrative tasks, based on the permissions associated with their roles, in any tenants to which they have administrative access. System configurations and limits have been set as well.

Using the APIs to set up a multitenant environment with system-level administration
The following operations show how you can set up a multitenant environment with system-level administration using Hybrid Data Pipeline APIs.

- Creating tenants with the Tenant API
- Retrieving roles with the Roles API
- Provisioning a user at the system level with the Tenant Administrator role
- Granting administrative access to the tenant with the Users API
- Granting administrative access to the tenant with the Tenant API
- Setting system configurations and limits
- Setting tenant limits
- Creating users and roles at the tenant level

Creating tenants with the Tenant API
In this example, a system administrator creates TenantA, TenantB, and TenantC using the Tenant API. The User (2) role has been specified as an imported role. As new tenants are created, the imported role becomes unique and is given a new ID.

Request to create TenantA

POST https://MyServer:8443/api/admin/tenants

Request Payload

```json
{
    "name": "TenantA",
    "description": "This is the HDP tenant for organization A.",
    "parentTenant": 1,
    "status": 1,
    "importedRoles": [
        2
    ]
}
```

Response Payload

```json
{
    "id": 61,
    "name": "TenantA",
    "description": "This is the HDP tenant for organization A.",
    "parentTenant": 1,
    "status": 1,
    "roles": [
        81
    ]
}
```
Request to create TenantB

POST https://MyServer:8443/api/admin/tenants

Request Payload

{
    "name": "TenantB",
    "description": "This is the HDP tenant for organization B.",
    "parentTenant": 1,
    "status": 1,
    "importedRoles": [2

}

Response Payload

{
    "id": 62,
    "name": "TenantB",
    "description": "This is the HDP tenant for organization B.",
    "parentTenant": 1,
    "status": 1,
    "roles": [82

}

Request to create TenantC

POST https://MyServer:8443/api/admin/tenants

Request Payload

{
    "name": "TenantC",
    "description": "This is the HDP tenant for organization C.",
    "parentTenant": 1,
    "status": 1,
    "importedRoles": [2

}

Response Payload

{
    "id": 63,
    "name": "TenantC",
    "description": "This is the HDP tenant for organization C.",
    "parentTenant": 1,
    "status": 1,
    "roles": [83

}

Retrieving roles with the Roles API

The system administrator must have the role ID to create a user with the Tenant Administrator role. The following GET operation retrieves the roles across the system.
Request

GET https://MyServer:8443/api/admin/roles

Response Payload

The first three roles in the payload are roles tied to the system tenant ("tenantId": 1). The remaining roles are the roles copied to the new tenants.

```
{
  "roles": [
    {
      "id": 1,
      "name": "System Administrator",
      "tenantId": 1,
      "description": "This role has all permissions. This role cannot be modified or deleted."
    },
    {
      "id": 2,
      "name": "User",
      "tenantId": 1,
      "description": "This role has the default permissions that a normal user will be expected to have."
    },
    {
      "id": 3,
      "name": "Tenant Administrator",
      "tenantId": 1,
      "description": "This role has all the tenant administrator permissions."
    },
    {
      "id": 81,
      "name": "User",
      "tenantId": 61,
      "description": "This role has the default permissions that a normal user will be expected to have."
    },
    {
      "id": 82,
      "name": "User",
      "tenantId": 62,
      "description": "This role has the default permissions that a normal user will be expected to have."
    },
    {
      "id": 83,
      "name": "User",
      "tenantId": 63,
      "description": "This role has the default permissions that a normal user will be expected to have."
    }
  ]
}
```

Creating a user at the system level with the Tenant Administrator role

With the following User API operation, the system administrator creates a user at the system level with the Tenant Administrator role. The tenant administrator must then be given administrative access to the tenant either through the Users API or the Tenant API, as described below.

Request

POST https://MyServer:8443/api/admin/users
Request Payload

```json
{
  "userName": "SysTenantAdmin",
  "tenantId": 1,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "password": "TempWord",
    "passwordStatus": 1,
    "passwordExpiration": null
  },
  "permissions": {
    "roles": [3]
  }
}
```

Response Payload

```json
{
  "id": 1001,
  "userName": "SysTenantAdmin",
  "tenantId": 1,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "passwordStatus": 1,
    "passwordExpiration": null
  },
  "permissions": {
    "roles": [3]
  },
  "authenticationInfo": {
    "authUsers": [
      {
        "authUserName": "SysTenantAdmin",
        "authServiceId": 1
      }
    ]
  }
}
```

**Granting administrative access to the tenant with the Users API**

In addition to user management permissions, a tenant administrator must be granted administrative access to the tenant. This can be done either through the Users API or the Tenant API. The following Users API request grants user account 2001 administrative access to TenantA (61).

**Request**

```
PUT https://MyServer:8443/api/admin/users/2001/tenantsadministered
```

**Request Payload**

```json
{
  "tenantsAdministered": [61]
}
```
Granting administrative access to the tenant with the Tenant API

In addition to user management permissions, a tenant administrator must be granted administrative access to the tenant. This can be done either through the Users API or the Tenant API. The following Tenant API request adds user account 2001 to the list of administrators who can administer the TenantA (61).

PUT https://MyServer:8443/api/admin/tenants/61

Request Payload

```json
{
    "admins": [
        391,
        502,
        2001
    ]
}
```

Response Payload

```json
{
    "admins": [
        391,
        502,
        2001
    ]
}
```

Setting system configurations and limits

Setting a system configuration

The following PUT operation disables IP address whitelists across all tenants. The number 8 is the ID of the IP address whitelist feature.

PUT https://MyServer:8443/api/admin/configurations/8

```json
{
    "value": "false"
}
```

Setting a system limit

The following POST creates a limit of 50000 concurrent OData queries across all tenants. The number 6 is the ID of the ODataMaxConcurrentQueries limit. The payload passes 50000 as the value for this limit.

POST https://MyServer:8443/api/admin/limits/system/6

```json
{
    "value": 50000
}
```
**Setting tenant limits**

The following POST creates a limit of 10000 concurrent OData queries on TenantA. The number 61 is the ID of TenantA, and the number 6 is the ID of the ODataMaxConcurrentQueries limit. This tenant limit will override the system limit.

```
POST https://MyServer:8443/api/admin/limits/tenants/61/6

{
  "value": 10000
}
```

**Creating users and roles at the tenant level**

The new system-level tenant administrator (SysTenantAdmin) can now provision users and create roles for TenantA, TenantB, and TenantC. The first request creates a new user in TenantA (61). The second request creates a new role in TenantA.

**Request to create user in TenantA**

```
POST https://MyServer:8443/api/admin/users

Request Payload

{
  "userName": "User1A",
  "tenantId": 61,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "password": "TempWord",
    "passwordStatus": 1,
    "passwordExpiration": null
  },
  "permissions": {
    "roles": [81]
  }
}
```

**Response Payload**

```
{
  "id": 2601,
  "userName": "User1A",
  "tenantId": 1,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "passwordStatus": 1,
    "passwordExpiration": null
  },
  "permissions": {
    "roles": [81]
  },
  "authenticationInfo": {
    "authUsers": [
      {"authUserName": "SysTenantAdmin",}...```
POST operation to create new role

With the following POST request, a new role is created in TenantA (61) for OData-only access to data sources. No user is specified in this example, but a user can subsequently be assigned the new role either through the Roles API or the Users API.

Request to create role in TenantA

POST https://MyServer:8443/api/admin/roles

Request Payload

```json
{
    "name": "ODataOnly",
    "tenantId": 61,
    "description": "This role allows only OData access.",
    "permissions": [7],
    "users": []
}
```

Response Payload

```json
{
    "id": 102,
    "name": "ODataOnly",
    "tenantId": 61,
    "description": "This role allows only OData access.",
    "permissions": [7],
    "users": []
}
```

See also

- User provisioning on page 107
- Users API on page 1146
- Roles API on page 1112
- System Configurations API on page 1124
- Limits API on page 1071

Setting up a multitenant environment with tenant-level administration

A system administrator can take the following general steps to set up a multitenant environment with tenant-level administration.

1. Create child tenants.
2. Create administrator roles in the child tenants.
3. Create tenant administrators who reside in the child tenants.
4. Set system configurations and limits.

See the following topics for details.

- Using the Web UI to set up a multitenant environment with tenant-level administration on page 99
Using the APIs to set up a multitenant environment with tenant-level administration on page 100

Using the Web UI to set up a multitenant environment with tenant-level administration

Take the following steps to set up a multitenant environment with tenant-level administration using the Web UI.

1. Create tenants.
   a) Navigate to the Manage Tenants view by clicking the manage tenants icon.
   b) Click + New Tenant.
   c) Under the General tab, enter a name and description for the tenant.
   d) Under the Roles tab, select any roles that you created in the system tenant that you want to import to the new tenant.
   e) Under the Limits tab, specify any limits that you want to set for the tenant. These limits will override limits at the system level.
   f) Click Save.

2. Create administrator roles.
   a) Navigate to the Manage Roles view by clicking the manage roles icon.
   b) For Tenant, select the child tenant for which you want to create the new administrator role.
   c) Click + New Role.
   d) Provide a name and description for the role.
   e) Select permissions to define the role.
   f) Click Save.

3. Create administrator users.
   a) Navigate to the Manage Users view by clicking the manage users icon.
   b) For Tenant, select the child tenant for which you want to create the new administrator user.
   c) Click + New User.
   d) Define the user with settings under each of the following tabs.
      • Under the General tab, enter a user name and assign the role you have created for the user.
      • Under the Authentication Setup tab, configure authentication settings.
      • Under the Limits tab, configure limits as desired. Note that user limits override system limits.
      • Under the Tenant Admin Access tab, grant the user administrative access to the child tenant.
   e) Click Save.

4. Set system configurations. These configurations apply to all tenants across the system.
a) Navigate to the **System Configurations** view by clicking the system configurations icon.
b) Configure options as desired. See **System Configurations view** on page 79 for option descriptions.
c) Click **Save**.

5. Set system limits.
   a) Navigate to the **Manage Limits** view by clicking the manage limits icon.
b) For **Tenant**, select **System** from the dropdown.
c) Set limits as desired. Limits set on tenants will override limits set at the system level.
d) Click **Save**.

**Results:**
You have created child tenants in the system tenant. In addition, you have created tenant administrators who reside in the child tenants. These administrators can perform administrative tasks, based on the permissions associated with their roles, in the tenants to which they belong and have administrative access. System configurations and limits have been set as well.

**Using the APIs to set up a multitenant environment with tenant-level administration**

The following operations show how you can set up a multitenant environment with tenant-level administration using Hybrid Data Pipeline APIs.

- Creating tenants with the Tenant API
- Retrieving roles with the Roles API
- Provisioning a tenant user with the Tenant Administrator role
- Granting administrative access to the tenant with the Users API
- Granting administrative access to the tenant with the Tenant API
- Setting system configurations and limits
- Setting tenant limits
- Creating users and roles at the tenant level

**Creating tenants with the Tenant API**

In this example, a system administrator creates the following tenants with the Tenant API: OrgA, OrgB, and OrgC. The default User (2) and Tenant Administrator (3) roles are being imported from the system tenant. As the system tenants are created, the imported roles becomes unique and are given a new IDs.

**Request to create OrgA**

```
POST https://MyServer:8443/api/admin/tenants
```

**Request Payload**

```
{
    "name": "OrgA",
    "description": "This is the HDP tenant for organization A.",
    "parentTenant": 1,
    "status": 1,
    "importedRoles": [
        2,
        3
    ]
}```
Response Payload

{
   "id": 71,
   "name": "OrgA",
   "description": "This is the HDP tenant for organization A.",
   "parentTenant": 1,
   "status": 1,
   "roles": [
      103,
      104
   ]
}

Request to create OrgB

POST https://MyServer:8443/api/admin/tenants

Request Payload

{
   "name": "OrgB",
   "description": "This is the HDP tenant for organization B.",
   "parentTenant": 1,
   "status": 1,
   "importedRoles": [
      2,
      3
   ]
}

Response Payload

{
   "id": 72,
   "name": "OrgA",
   "description": "This is the HDP tenant for organization A.",
   "parentTenant": 1,
   "status": 1,
   "roles": [
      105,
      106
   ]
}

Request

POST https://MyServer:8443/api/admin/tenants

Request Payload to create OrgC

{
   "name": "OrgC",
   "description": "This is the HDP tenant for organization C.",
   "parentTenant": 1,
   "status": 1,
   "importedRoles": [
      2,
      3
   ]
}
Response Payload

{
  "id": 73,
  "name": "OrgC",
  "description": "This is the HDP tenant for organization C.",
  "parentTenant": 1,
  "status": 1,
  "roles": [
    107,
    108
  ]
}

Retrieving roles with the Roles API

The system administrator must have the role ID to create a user with the Tenant Administrator role. The following GET operation retrieves the roles across the system.

Request

GET https://MyServer:8443/api/admin/roles

Note: The ?tenantID=<tenant_id> and ?tenantName=<tenant_name> query parameters can be appended to the URL to limit the roles returned to a specific tenant.

Response Payload

The first three roles in the payload are roles tied to the system tenant ("tenantId": 1). The remaining roles are the User and Tenant Administrator roles copied to the new tenants.

{
  "roles": [
    {
      "id": 1,
      "name": "System Administrator",
      "tenantId": 1,
      "description": "This role has all permissions. This role cannot be modified or deleted."
    },
    {
      "id": 2,
      "name": "User",
      "tenantId": 1,
      "description": "This role has the default permissions that a normal user will be expected to have."
    },
    {
      "id": 3,
      "name": "Tenant Administrator",
      "tenantId": 1,
      "description": "This role has all the tenant administrator permissions."
    },
    {
      "id": 103,
      "name": "User",
      "tenantId": 71,
      "description": "This role has the default permissions that a normal user will be expected to have."
    },
    {
      "id": 104,
      "name": "Tenant Administrator",
      "tenantId": 71,
      "description": "This role has all the tenant administrator permissions."
    }
  ]
}
Provisioning a tenant user with the Tenant Administrator role

With the following User API operation, the system administrator creates a user in the OrgA tenant (71) with the Tenant Administrator role. The tenant administrator must then be given administrative access to the tenant either through the Users API or the Tenant API, as described below.

Request

POST https://MyServer:8443/api/admin/users

Request Payload

{
"userName": "OrgAAdmin",
"tenantId": 71,
"statusInfo": {
"status": 1,
"accountLocked": false
},
"passwordInfo": {
"password": "TempWord",
"passwordStatus": 1,
"passwordExpiration": null
},
"permissions": {
"roles": [103, 104]
}
}

Response Payload

{
"id": 2001,
"userName": "OrgAAdmin",
}
Granting administrative access to the tenant with the Users API

In addition to user management permissions, a tenant administrator must be granted administrative access to the tenant. This can be done either through the Users API or the Tenant API. The following Users API request grants user account 2001 administrative access to the OrgA tenant (71).

**Request**

```
PUT https://MyServer:8443/api/admin/users/2001/tenantsadministered
```

**Request Payload**

```
{
   "tenantsAdministered": [
      71
   ]
}
```

**Response Payload**

```
{
   "tenantsAdministered": [
      71
   ]
}
```

Granting administrative access to the tenant with the Tenant API

In addition to user management permissions, a tenant administrator must be granted administrative access to the tenant. This can be done either through the Users API or the Tenant API. The following Tenant API request adds user account 2001 to the list of administrators who can administer the OrgA tenant (71).

**Request**

```
PUT https://MyServer:8443/api/admin/tenants/71
```

**Request Payload**

```
{
   "admins": [
      391,
      103,
      104
   ]
}
```
Response Payload

{
   "admins": [
      391,
      502,
      2001
   ]
}

Setting system configurations and limits

Setting a system configuration

The following PUT operation disables IP address whitelists across all tenants. The number 8 is the ID of the IP address whitelist feature.

PUT https://MyServer:8443/api/admin/configurations/8

{
   "value": "false"
}

Setting a system limit

The following POST creates a limit of 50000 concurrent ODdata queries across all tenants. The number 6 is the ID of the ODataMaxConcurrentQueries limit. The payload passes 50000 as the value for this limit.

POST https://MyServer:8443/api/admin/limits/system/6

{
   "value": 50000
}

Setting tenant limits

The following POST creates a limit of 10000 concurrent ODdata queries on the OrgA tenant. The number 71 is the ID of OrgA, and the number 6 is the ID of the ODataMaxConcurrentQueries limit. This tenant limit will override the system limit.

POST https://MyServer:8443/api/admin/limits/tenants/71/6

{
   "value": 10000
}

Creating users and roles at the tenant level

The new tenant administrator (OrgAAdmin) can now provision users and create roles for the OrgA tenant (71). The first request creates a new user in OrgA. The second request creates a new role in OrgA.

Request

POST https://MyServer:8443/api/admin/users
Request Payload

```json
{
    "userName": "OrgAUser1",
    "tenantId": 71,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "password": "TempWord",
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [104]
    }
}
```

Response Payload

```json
{
    "id": 3222,
    "userName": "OrgAUser1",
    "tenantId": 71,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [104]
    },
    "authenticationInfo": {
        "authUsers": [
            {
                "authUserName": "OrgAUser1",
                "authServiceId": 1
            }
        ]
    }
}
```

With the following POST request, a new role is created in the OrgA tenant for OData-only access to data sources. No user is specified in this example, but a user can subsequently be assigned the new role either through the Roles API or the Users API.

**Request**

```
POST https://MyServer:8443/api/admin/roles
```

**Request Payload**

```json
{
    "name": "ODataOnly",
    "tenantId": 71,
    "description": "This role allows only OData access.",
    "permissions": [7],
    "users": []
}
```
Response Payload

```
{
    "id": 311,
    "name": "ODataOnly",
    "tenantId": 71,
    "description": "This role allows only OData access.",
    "permissions": [
    7
    ],
    "users": []
}
```

See also

- User provisioning on page 107
- Users API on page 1146
- Roles API on page 1112
- System Configurations API on page 1124
- Limits API on page 1071

User provisioning

Once a tenant architecture has been established as described in Tenant architectures on page 82, a Hybrid Data Pipeline administrator can proceed with provisioning users. User accounts can be created and managed either through the Web UI or using Hybrid Data Pipeline APIs. User accounts must have at least one assigned role. A role is defined by the permissions that are associated with it. Users can be provisioned to have either direct access to the Hybrid Data Pipeline service or query-only access to Hybrid Data Pipeline data sources. Whether a user is a direct-access or query-only user depends on the role assigned and its associated permissions.

- Direct-access user
- Query-only user
- Administrator permissions
- User provisioning scenarios

Direct-access user

A direct-access user is a user the administrator has provisioned with direct access to the service to create, manage, and query data sources. The following work flow describes how access to data may be established with a direct-access user.

1. The administrator creates a role for a direct-access user.
2. The administrator creates a user account for the direct-access user.
3. The direct-access user creates a data source through either the Web UI or the Data Sources API on page 1277.

**Note:** Alternatively, administrators can create their own data sources and share them with users or create data sources on behalf of users.

4. Data source connection information is integrated into a client-side application or BI tool.
Query-only user

An administrator can limit user access such that users can run applications against Hybrid Data Pipeline data sources, but not access Hybrid Data Pipeline directly. In this scenario, the administrator must not only provision user accounts, but also create the data sources against which queries will be made. The data source information may then be supplied either directly to the query-only user, or integrated into the client application such that data access is transparent to the application end user. Thus, client applications are given access to backend data stores, while users and developers on the client side do not otherwise have access to Hybrid Data Pipeline.

When provisioning users for query-only access to Hybrid Data Pipeline data sources, administrators can manage data sources in two distinct ways.

• First, they can create a data source themselves, and then share the data source with one or more user accounts. In this case, the data source information, including connection information, is the same for all accounts querying the data source. Hence, sharing data sources can be used to support general access to a backend data store when access to the data is the same across multiple end users. For example, an administrator might create a data source to support the use of a reporting tool. Multiple end users across the organization use the tool to run reports against the backend data store. In this case, connection information associated with the data source can be integrated with the reporting tool. Hybrid Data Pipeline may be entirely transparent to the users running the reports. However, the reporting tool uses the Hybrid Data Pipeline data source to access the backend data.

• Second, the administrator can create a data source on behalf of a user account. In this scenario, the data source is owned by the user account, and the data source information is unique to the account. Therefore, creating data sources on behalf of users should be used in scenarios where access to backend data must be unique for each user. For example, a backend data store might have row-level security measures on an Employee database such that managers are only able to access information for the employees they manage. In this case, an administrator would create data sources on the backend data store that are unique to each manager based on each manager's credentials.

Note: At this time, administrators must use the Hybrid Data Pipeline API to share data sources or create data sources on behalf of users. Neither of these features can be implemented through the Web UI.

The following workflow describes how access to data may be enabled for a query-only user.

1. The administrator creates a role for the query-only user.
2. The administrator creates a user account for the query-only user.
3. The administrator uses either of the following methods to create a Hybrid Data Pipeline data source for the query-only user.
   a. The administrator creates a data source through either the Web UI or the Data Sources API on page 1277. The administrator then shares the data source with the query-only user as described in Sharing data sources on page 1279.
   b. The administrator creates a data source on behalf of the query-only user as described in the Data Sources API on page 1277 and Managing resources on behalf of users on page 1281.
4. Data source connection information is integrated into a client-side application or BI tool.

Administrator permissions

The ability of an administrator to provision users depend on the administrator's permissions and administrative access to a given tenant. A system administrator – defined as a user with the Administrator (12) permission – can provision users across any tenant in the system. An administrator who does not have the Administrator (12) permission must meet the following requirements to provision users.

• WebUI (8) permission must be granted if the administrator is using the Web UI to provision users.
• Administrative access to the tenant. In the Web UI, administrative access to a tenant can be granted by editing a user account via the Manage Users view on page 65. With the API, administrative access can be granted either by updating the tenants administered for a user via the Users API or by updating the list of administrators for a tenant via the Tenant API.

• The permission corresponding to the specific operation. For example, the administrator must have the CreateUser (13) permission to create a user account, or the DeleteUsers (16) permission to delete a user account.

User provisioning scenarios
The following topics describe a number of Hybrid Data Pipeline user provisioning scenarios.

• Provisioning users with the Web UI on page 109
• Provisioning users with Hybrid Data Pipeline APIs on page 114
• Managing permissions with Hybrid Data Pipeline APIs on page 133

Provisioning users with the Web UI
The Web UI can be used to provision and manage Hybrid Data Pipeline user accounts. The Web UI can also be used to create, view, modify, and delete roles, and, more generally, manage roles and the users associated with them. Depending on the role assigned and its associated permissions, users may have either direct access to the service or query-only access to data sources.

The following topics provide instructions for provisioning users with the Web UI. (See also Using the Web UI on page 63.)

• Create user accounts on page 109
• Update user accounts on page 110
• Delete user accounts on page 111
• Create roles on page 112
• Update roles on page 112
• Delete roles on page 113
• View data sources or data source groups on page 113
• Reset user account password on page 113

Create user accounts
Take the following steps to create a user account through the Web UI.

1. Navigate to the Manage Users view by clicking the manage users icon.
2. Click + New User.
3. Under the General tab, provide the following information.

   • Tenant. Select the tenant to which the user will belong. Only the tenants for which you have administrative access will appear.

   • User Name. Enter the name of the user.

   • Role. Assign a role for the user. A role must be assigned for the user.
• **Status.** Specify the user's status. The user can be active or inactive.

4. Under the **Authentication Setup** tab, specify the **Authentication Type**.
   
   • If you select **Internal**, you must specify a username and password.
   
   • If you select an external authentication service, you must specify one or more users whose credentials are maintained by the service. This action associates the user with the Hybrid Data Pipeline user account.

   **Note:** See **Authentication** on page 143 for information on implementing authentication services.

5. Under the **Limits** tab, set throttling limits for the user as desired.

   • **MaxFetchRows.** The maximum number of rows to be fetched for a single query.
   
   • **ODataMaxConcurrentQueries.** The maximum number of concurrent active OData queries per data source.
   
   • **TransactionTimeout.** The number of seconds the system allows a transaction to be idle before rolling it back.
   
   • **XdbcMaxResponse.** The approximate maximum size of JDBC/ODBC HTTP result data (in KB).

6. Under the **Tenant Admin Access** tab, administrative access to tenants may be granted if desired.

7. Click **Save**.

**Results:**

The user has been created. The user will appear in the list of users in the **Manage Users** view for the given tenant.

**What to do next:**

Depending on the application environment, either of the following actions may be taken.

• The direct-access user creates a data source through either the **Web UI** or the **Data Sources API** on page 1277.

• The administrator creates a data source through either the **Web UI** or the **Data Sources API** on page 1277. The administrator then shares the data source with the user as described in **Sharing data sources** on page 1279.

• The administrator creates a data source on behalf of the user as described in the **Data Sources API** on page 1277 and **Managing resources on behalf of users** on page 1281.

**Update user accounts**

The **Web UI** can be used to update the following attributes of a user account. 

• User name

• User role

• User status (active or inactive)

• Authentication service

• User password

• Throttling limits
• Administrative access

Take the following steps to update a user account through the Web UI.

1. Navigate to the Manage Users view by clicking the manage users icon.
2. Select the user account you want to update.
3. Click the Actions dropdown. Then select Edit.
4. Under the General tab, update user information as desired.
   - **Tenant.** The tenant field displays the tenant to which the user belongs. However, you may not select another tenant in order to transfer the user. To transfer a user from one tenant to another, you must create a new user account in the tenant to which the user is moving.
   - **User Name.** You may edit the user name field.
   - **Role.** You may assign a different or additional role to the user.
   - **Status.** You may change the user's status. The user can be active or inactive.
5. Under the Authentication Setup tab, update authentication information as desired.
   - **Authentication Type.** Specify the method of authentication the user must use to login. In addition to the internal authentication service, an administrator can integrate an external authentication service such as Active Directory. See Authentication on page 143 for details.
   - **Password.** Enter a new user password. See Password policy on page 159 for password requirements.
   - **Confirm Password.** Re-enter the new password.
6. Under the Limits tab, update throttling limits for the user as desired.
   - **MaxFetchRows.** The maximum number of rows to be fetched for a single query.
   - **ODataMaxConcurrentQueries.** The maximum number of concurrent active OData queries per data source.
   - **TransactionTimeout.** The number of seconds the system allows a transaction to be idle before rolling it back.
   - **XdbcMaxResponse.** The approximate maximum size of JDBC/ODBC HTTP result data (in KB).
7. Under the Tenant Admin Access tab, administrative access to tenants may be granted or removed.
8. Click Save.

**Results:**
The user has been updated.

**Delete user accounts**
Take the following steps to delete user account through the Web UI.

1. Navigate to the Manage Users view by clicking the manage users icon.
2. Select the user account(s) you want to delete.
3. Click the **Actions** dropdown. Then select **Delete**.

4. When prompted confirm that you wish to delete the user(s) by clicking **Delete**.

**Results:**
The user(s) has been delete. The user(s) is removed from the list of users in the **Manage Users** view for the given tenant.

---

**Create roles**
Take the following steps to create a role through the Web UI.

1. Navigate to the **Manage Roles** view by clicking the manage roles icon.
2. Click **+ New Role**.
3. Provide the following information.
   - **Tenant**: Select the tenant for which the role is being created. Only the tenants for which you have administrative access will appear.
   - **Role Name**: Enter the name of the role.
   - **Role Description**: Enter a description of the role.
   - **Permissions**: Select the permissions associated with the role. See Permissions and default roles on page 59 for details.
4. Click **Save**.

**Results:**
The role has been created. The role will appear in the list of roles in the **Manage Roles** view for the given tenant.

**What to do next:**
You can now create users with this role, or assign this role to users.

---

**Update roles**
Take the following steps to update a role through the Web UI.

1. Navigate to the **Manage Roles** view by clicking the manage roles icon.
2. Select the role you want to update.
3. Click the **Actions** dropdown. Then select **Edit**.
4. Update the role as desired.
   - **Tenant**: The tenant field displays the tenant to which the role belongs. However, you may not select another tenant in order to transfer the role. A distinct role must be created for the other tenant.
   - **Role Name**: You may enter a new name for the role.
   - **Role Description**: You may enter a new description for the role.
• **Permissions.** You may modify the permissions associated with the role. See Permissions and default roles on page 59 for details.

5. Click **Save**.

**Results:**
The role has been updated. The permissions for the users to whom the role was assigned are modified accordingly.

**Delete roles**
Take the following steps to delete roles through the Web UI.

1. Navigate to the **Manage Roles** view by clicking the manage roles icon.
2. Select the role(s) you want to delete.
3. Click the **Actions** dropdown. Then select **Delete**.
4. When prompted confirm that you wish to delete the roles(s) by clicking **Delete**.

**Results:**
The role(s) has been delete. The role(s) is removed from the list of roles in the **Manage Roles** view for the given tenant.

**View data sources or data source groups**
In the tenants they administer, administrators can view a list of data sources owned by the users that reside in the tenant.
Take the following steps to view data sources in the tenant.

1. Navigate to the **Data Sources** view by clicking the data sources icon. By default, a list of data sources owned by the administrator will be shown.
2. Specify whether you want to view the user's data sources or the user's data source groups.
   • Select the **Data Sources** tab to view the user's data sources.
   • Select the **Data Source Groups** tab to view the user's data source groups.
3. Select the user's tenant and then the user's name from the **Select Tenant** and **Select User** dropdowns.

**Results:**
A list of data sources or data source groups owned by the user is displayed.

**Reset user account password**
Take the following steps to reset a user account password through the Web UI.

1. Navigate to the **Manage Users** view by clicking the manage users icon.
2. Select the user account you want to update.
3. Click the **Actions** dropdown. Then select **Edit**.
4. Select the **Authentication Setup** tab.
5. Enter a password in the **Password** field. See **Password policy** on page 159 for password requirements.
6. Re-enter the password in the **Confirm Password** field.
7. Click **Save**.

**Results:**
The user password has been reset.

---

**Provisioning users with Hybrid Data Pipeline APIs**

Administrators can use Hybrid Data Pipeline APIs to provision users for direct access to Hybrid Data Pipeline. The **Users API** on page 1146 can be used to provision and manage Hybrid Data Pipeline user accounts. The **Roles API** on page 1112 can be used to create, view, modify, and delete roles, and, more generally, manage roles and the users associated with them.

The following topics detail API operations for provisioning users in a number of scenarios. (See also the **Hybrid Data Pipeline API reference** on page 1037.)

- Providing direct access on page 114
- Providing query-only access by sharing a data source on page 119
- Providing query-only access by creating data sources on behalf of users on page 122
- Providing limited direct access to data sources and features on page 125
- Providing query access to an ODBC data source and limited access to the Web UI on page 129

**Providing direct access**
The following operations show how you can provision a **direct-access user** with Hybrid Data Pipeline APIs.

- Creating a user account
- Creating new role
- Assigning new role
- Setting permissions on a user account
- Resetting user account password
- Changing user account status
- Deleting a user account

**Creating a user account**
The following operation creates a user account in tenant 26 with role 86. The administrator must have the Administrator (12) permission, or the CreateUsers (13) permission and administrative access on the tenant.

**Note:** An administrator cannot create users that have tenant or elevated permissions unless the administrator also has those permissions.

**Request**

POST https://MyServer:8443/api/admin/users
Request Payload

```
{
    "userName": "testuser",
    "tenantId": 26,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "password": "TempPassword",
        "passwordStatus": 1,
        "passwordExpiration": "2020-01-01 00:00:00"
    },
    "permissions": {
        "roles": [86]
    }
}
```

Response Payload

```
{
    "id": 31,
    "userName": "testuser",
    "tenantId": 26,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "password": "TempPassword",
        "passwordStatus": 1,
        "passwordExpiration": "2020-01-01 00:00:00"
    },
    "permissions": {
        "roles": [86]
    },
    "authenticationInfo": {
        "authUsers": [
            {
                "authUserName": "testuser",
                "authServiceId": 1
            }
        ]
    }
}
```

Creating new role

The following operation creates a new role in tenant 26. The administrator must have the Administrator (12) permission; or the administrator must have the CreateRole (17) permission, any permissions specified in the new role, and administrative access on the tenant.

Request

```
POST https://MyServer:8443/api/admin/roles
```

Request Payload

```
{
    "name": "odata_ds_role",
    "tenantId": 26,
    "description": "This role allows users to create and work with OData data sources."
}
```
"permissions": [1, 2, 3, 4, 7, 8, 9, 10, 11],
"users": []
}

Response Payload
{
"id": 94,
"name": "odata_ds_role",
"tenantId": 26,
"description": "This role allows users to create and work with OData data sources.",
"permissions": [1, 2, 3, 4, 7, 8, 9, 10, 11],
"users": []
}

Assigning new role
The following operation assigns the odata_ds_role to the testuser user account. The user account ID 31 is specified in the URL. The administrator must have the Administrator (12) permission; or the administrator must have the ModifyUsers (15) permission, any permissions specified in the new role, and administrative access on the tenant.

Request
PUT https://MyServer:8443/api/admin/users/31

Request Payload
{
"userName": "testuser",
"tenantId": 26,
"statusInfo": {
"status": 1,
"accountLocked": false
},
"passwordInfo": {
"passwordStatus": 1,
"passwordExpiration": "2025-01-01 00:00:00"
},
"permissions": {
"roles": [94]}}
Response Payload

{
  "userName": "testuser",
  "tenantId": 26,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "passwordStatus": 1,
    "passwordExpiration": "2025-01-01 00:00:00"
  },
  "permissions": {
    "roles": [
      94
    ],
    "permissions": [
      5
    ]
  },
  "authenticationInfo": {
    "authUsers": [
      {
        "authUserName": "testuser",
        "authServiceId": 1
      }
    ]
  }
}

Setting permissions on a user account

The following operation shows how permissions can be set explicitly on a user account. In this example, the administrator retains the odata_ds_role for the user, but adds the UseDataSourceWithJDBC (5) permission. The administrator must have the Administrator (12) permission; or the administrator must have the ModifyUsers (15) permission, any permissions specified in the new role, and administrative access on the tenant.

Request

PUT https://MyServer:8443/api/admin/users/31/permissions

Request Payload

{
  "roles": [
    94
  ],
  "permissions": [
    5
  ]
}

Response Payload

{
  "roles": [
    94
  ],
  "permissions": [
    5
  ]
}
Resetting user account password

The following operation shows how to reset a user account password. Making this request changes the password and sets the `passwordStatus` to 2 (reset). The end user must change the password when he or she next logs in. Users can change their passwords either through the Web UI or through the User Details API. The administrator must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.

Request

```
PUT https://MyServer:8443/api/admin/users/31/resetpassword
```

Request Payload

```
{
   "newPassword": "tempsecret"
}
```

Response Payload

```
Status code: 204
No Content
```

Changing user account status

The following operation shows how to change user account status from active (1) to inactive (0). An inactive user cannot log into the Web UI, use APIs, or establish JDBC, ODBC, or OData connections. The administrator must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.

Request

```
PUT https://MyServer:8443/api/admin/users/31/statusinfo
```

Request Payload

```
{
   "status": 0
}
```

Response Payload

```
{
   "status": 0
}
```

Deleting user account

The following operation shows how to delete a user account. The user account ID 31 is specified in the URL. The administrator must have the Administrator (12) permission, or the DeleteUsers (16) permission and administrative access on the tenant.

Request

```
DELETE https://MyServer:8443/api/admin/users/31
```
Providing query-only access by sharing a data source

The following operations show the provisioning of a query-only user for ODBC access to a SQL Server database. The administrator begins by creating a role for the user account, creates a user account, creates a data source, and then shares the data source with the user account.

Note: A data source can also be shared with a tenant, in effect sharing the data source with all the users in the tenant. See Sharing data sources on page 1279 for details.

- Create role for query-only access
- Create user account
- Create data source
- Share data source

Create role for query-only access

The administrator begins by creating a role for query-only access with the following operation. The administrator must have the Administrator (12) permission, or the CreateRole (17) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/roles

Request Payload

```json
{
    "name": "Query access",
    "tenantId": 53,
    "description": "This role permits only query access.",
    "permissions": [5, 6, 7],
    "users": []
}
```

Response Payload

```json
{
    "id": 62,
    "name": "Query access",
    "tenantId": 59,
    "description": "This role permits only query access.",
    "permissions": [5, 6, 7],
    "users": []
}
```
Create user account

The administrator then provisions a user account with the "Query access" role. The administrator must have the Administrator (12) permission, or the CreateUser (13) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/users

Request Payload

```json
{
    "userName": "QueryOnlyUser",
    "tenantId": 59,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "password": "TempPassword",
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [44]
    }
}
```

Response Payload

```json
{
    "id": 921,
    "userName": "QueryOnlyUser",
    "tenantId": 56,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [44]
    },
    "authenticationInfo": {
        "authUsers": [
            {
                "authUserName": "QueryOnlyUser",
                "authServiceId": 1
            }
        ]
    }
}
```

Create a data source

The administrator then creates a data source. The administrator will be the owner of this data source, but will share the data source with ODBCUser in the next operation.

The administrator must have the Administrator (12) permission, or the MgmtAPI (11) and CreateDataSource (1) permissions.
Request

POST https://MyServer:8443/api/mgmt/datasources

Request Payload

{
    "name": "SQLServer2",
    "dataStore": "46",
    "connectionType": "Hybrid",
    "description": "Test SQL Server access",
    "options": {
        "Database": "CustomerData",
        "User": "MySQLServerUserId",
        "Password": "MySQLServerPassword"
    }
}

Response Payload

{
    "id": "6334",
    "name": "SQLServer2",
    "dataStore": "46",
    "connectionType": "Hybrid",
    "description": "Test SQL Server access",
    "options": {
        "Database": "CustomerData",
        "User": "MySQLServerUserId",
        "Password": "MySQLServerPassword"
    }
}

Share a data source

The administrator then shares the data source with the QueryOnlyUser. The administrator limits access to
ODBC-only queries by setting the UseDataSourceWithODBC (6) permission on the data source. The data
source ID 6334 is passed in the request URL, while the user ID 921 and the data source permission are passed
in the request payload.

The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI
(11) permission, the ModifyDataSource (3) permission, the UseDataSourceWithODBC (6) permission, and
administrative access to the tenant to which the shared user belongs.

Request

POST https://MyServer:8443/api/mgmt/datasources/6334/sharedUsers

Request Payload

{
    "sharedUsers": [
    {
        "userId": 921,
        "permissions": [6
    ]
    ]
}
Response Payload

Status code: 201
Successful response

{
  "sharedUsers": [
    {
      "userId": 921,
      "permissions": [6]
    }
  ]
}

Providing query-only access by creating data sources on behalf of users

The following operations show the provisioning of a query-only user for OData access to an Oracle database. The administrator begins by creating a role for the user account, next creates the user account, and then creates a data source on behalf of the user. (See also Managing resources on behalf of users on page 1281.)

- Create role for OData query-only access
- Create user account
- Create a data source on behalf of the user account
- Retrieve data source information on behalf of the user account
- User queries the OData endpoint

Create role for OData query-only access

The administrator begins by creating a role for OData query-only access with the following operation. The administrator must have the Administrator (12) permission, or the CreateRole (17) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/roles

Request Payload

{
  "name": "OData query",
  "tenantId": 56,
  "description": "This role permits only OData query access.",
  "permissions": [
    7
  ],
  "users": []
}

Response Payload

{
  "id": 21,
  "name": "OData-only Users",
  "tenantId": 56,
  "description": "This role permits only OData query access.",
  "permissions": [
    7
  ],
  "users": []
}
Create user account

The administrator then provisions a user account with the "OData query" role. The administrator must have the Administrator (12) permission, or the CreateUsers (13) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/users

Request Payload

```
{
    "userName": "ODataUser",
    "tenantId": 56,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "password": "TempPassword",
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [21]
    }
}
```

Response Payload

```
{
    "id": 921,
    "userName": "ODataUser",
    "tenantId": 56,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [21]
    },
    "authenticationInfo": {
        "authUsers": [
            {
                "authUserName": "ODataUser",
                "authServiceId": 1
            }
        ]
    }
}
```

Create a data source on behalf of the user account

The administrator then creates a data source on behalf of ODataUser. Since the only permission associated with the assigned role is UseDataSourceWithOData (7), the user will be able to access data through this data source with OData queries, but will not be able to view data source information or access other Hybrid Data Pipeline features.
The user query parameter (?user) is used to specify the owner of the data source. The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the CreateDataSource (1) permission.

Request


Request Payload

{
  "name": "Oracle_OData",
  "dataStore": 43,
  "connectionType": "Hybrid",
  "description": "",
  "options": {
    "User": "OracleTest",
    "Password": "Secret",
    "ODataSchemaMap": "{"odata_mapping_v2":{"schemas":
  
  
  
  
  
  "tables":
  
  
  
  
  
  "ServerName": "TestServer",
    "ExtendedOptions": "EncryptionMethod=noEncryption",
    "SID": "UNI",
    "ODataVersion": "2"
  }
}

Response Payload

{
  "id": "1681",
  "name": "Oracle_OData",
  "dataStore": 43,
  "connectionType": "Hybrid",
  "description": "",
  "options": {
    "User": "OracleTest",
    "Password": "Secret",
    "ODataSchemaMap": "{"odata_mapping_v2":{"schemas":
  
  
  
  
  
  "tables":
  
  
  
  
  
  "ServerName": "TestServer",
    "ExtendedOptions": "EncryptionMethod=noEncryption",
    "SID": "UNI",
    "ODataVersion": "2"
  }

Retrieve data source information on behalf of the user account

The administrator can then retrieve data source details on behalf of ODataUser. The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the ViewDataSource (2) permission. (Note that ODataUser cannot retrieve this information because the user does not have ViewDataSource (2) permission.)

Request

Response Payload

```json
{
  "id": "1681",
  "name": "Oracle_OData",
  "dataStore": 43,
  "connectionType": "Hybrid",
  "description": "",
  "options": {
    "User": "OracleTest",
    "Password": "Secret",
    "ODataSchemaMap": "{\"odata_mapping_v2\":{\"schemas\":[{\"name\":\"D2CQA01\",\"tables\":\"Dept_Emp\":{},\"Employees\":{},\"Departments\":{},\"Salaries\":{},\"Titles\":{},\"Dept_Manager\":{}]},\"ServerName\":\"TestServer\",\"ExtendedOptions": \"EncryptionMethod=noEncryption\",\"SID": \"UNI\",\"ODataVersion": "2\"}}
  }
}
```

User queries the OData endpoint

With the appropriate connection information as supplied by the administrator, the ODataUser can now query the OData endpoint. With the following request, ODataUser retrieves an XML document from the Oracle_OData data source.

**Important:** The new user must authenticate using basic authentication to execute API queries.

Request

```
GET https://MyServer:8443/api/odata/Oracle_OData/Employees
```

Response Payload

```xml
<?xml version='1.0' encoding='utf-8'?>
  <title type="text">Employees</title>
  <id>https://MyServer:8443/api/odata/Oracle_OData/Oracle_OData/Employees</id>
  <updated>2018-03-29T17:58:44Z</updated>
  <link rel="self" title="Employees" href="Employeeses"/>
  <entry>
    <id>https://MyServer:8443/api/odata/Oracle_OData/Oracle_OData/Employees(10001M)</id>
    <title type="text"/>
    <updated>2018-03-29T17:58:44Z</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Employees" href="Employeeses(10001M)"/>
    ...
  </entry>

Providing limited direct access to data sources and features

The following operations show the provisioning of a direct-access user. The user is granted permission to query data sources and use a number of features, including the Web UI, but is not granted permission to create, view, or modify data sources.

- Create query-based role
• Create SQL user
• Create a data source
• Share data source with SQLUser

Create query-based role

With the following request, an administrator can create a role that gives a user permissions to query OData, ODBC, and JDBC data sources. In addition, the user has access to the Web UI, can change their password in the Web UI, and can query data sources they own using the SQL Editor. However, the role does not permit the user to create, modify, or delete data sources. The administrator must have the Administrator (12) permission, or the CreateRole (17) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/roles

Request Payload

```json
{
    "name": "QueryBasedRole",
    "tenantId": 56,
    "description": "This role allows query access and direct access for the Web UI, password, SQL editor, and Management API features",
    "permissions": [5, 6, 7, 8, 9, 10, 11],
    "users": []
}
```

Response Payload

```json
{
    "id": 88,
    "name": "QueryBasedRole",
    "tenantId": 56,
    "description": "This role allows query access and direct access for the Web UI, password, SQL editor, and Management API features",
    "permissions": [5, 6, 7, 8, 9, 10, 11],
    "users": []
}
```

Create SQL user

With the following request, an administrator creates a user called SQLUser with the QueryBasedRole role. SQLUser inherits the permissions of the QueryBasedRole role described above. The administrator must have the Administrator (12) permission, or the CreateUsers (13) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/users

Request Payload

```json
{
    "userName": "SQLUser",
```
Create a data source

An administrator can then create a data source. The administrator will be the owner of this data source, but will share the data source with SQLUser in the next operation.

The administrator must have the Administrator (12) permission, or the MgmtAPI (11) and CreateDataSource (1) permissions.

Request

POST https://MyServer:8443/api/mgmt/datasources

Request Payload

{  "name": "Oracle_Test",  "dataStore": 43,  "connectionType": "Hybrid",  "description": "",  "options": {    "User": "Test",    "Password": "Test",  }"tenantId": 56,  "statusInfo": {    "status": 1,    "accountLocked": false  },  "passwordInfo": {    "password": "Secret",    "passwordStatus": 1  },  "permissions": {    "roles": [88]  },  }

Response Payload

{  "id": 1297,  "userName": "SQLUser",  "tenantId": 56,  "statusInfo": {    "status": 1,    "accountLocked": false  },  "passwordInfo": {    "passwordStatus": 1,    "passwordExpiration": null  },  "permissions": {    "roles": [88]  },  "authenticationInfo": {    "authUsers": [      {        "authUserName": "SQLUser",        "authServiceId": 1      }    ]  }}
Response Payload

```
{
    "id": "13",
    "name": "Oracle_Test",
    "dataStore": 43,
    "connectionType": "Hybrid",
    "description": "",
    "options": {
        "User": "Test",
        "Password": "Test",
        "ServerName": "OracleTest",
        "ODataSchemaMap": "{{odata_mapping_v2":{"schemas":{"name":"D2CQA01","tables":{"Dept_Emp":{},"Employees":{},"Departments":{},"Salaries":{},"Titles":{},"Dept_Manager":{}}}}},
        "ODataVersion": "2",
        "SID": "UNI",
        "ExtendedOptions": "EncryptionMethod=noEncryption"
    }
}
```

**Share a data source**

The administrator can then share the data source with the SQLUser. The administrator limits access to queries by setting the UseDataSourceWithJDBC (5), UseDataSourceWithODBC (6), and UseDataSourceWithOData (7) permissions on the data source. The data source ID 13 is passed in the request URL, while the user ID 1297 and the data source permission are passed in the request payload.

The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the ModifyDataSource (3) permission, the query permissions, and administrative access to the tenant to which the shared user belongs.

**Request**

POST https://MyServer:8443/api/mgmt/datasources/13/sharedUsers

**Request Payload**

```
{
    "sharedUsers": [
        {
            "userId": 1297,
            "permissions": [5, 6, 7]
        }
    ]
}
```

**Response Payload**

Status code: 201
Successful response

```
{
    "sharedUsers": [
```
Providing query access to an ODBC data source and limited access to the Web UI

The following operations show the provisioning of a direct-access user. The user is initially granted access to query ODBC data sources and to change their password via the Web UI. Then, the user is subsequently granted access to the SQL Editor.

- Create role for ODBC-only user with access to change password in the Web UI
- Create ODBC-only user
- Create a data source on behalf of ODBC-only user
- Update ODBC-only role to include SQL Editor access
- Grant SQL Editor access explicitly to the ODBC-only user

Create role for ODBC-only user with access to change password in the Web UI

With the following request, an administrator can create a role for an ODBC-only user with Web UI access to change their password. The administrator must have the Administrator (12) permission, or the CreateRole (17) permission and administrative access on the tenant.

**Note:** To use change password functionality in the Web UI, Web UI permission must also be granted.

**Request**

```
POST https://MyServer:8443/api/admin/roles
```

**Request Payload**

```
{
   "name": "ODBC-only Users",
   "tenantId": 56,
   "description": "This role has ODBC, WebUI, and change password permissions.",
   "permissions": [6, 8, 9],
   "users": []
}
```

**Response Payload**

```
{
   "id": 42,
   "name": "ODBC-only Users",
   "tenantId": 56,
   "description": "This role has ODBC, WebUI, and change password permissions.",
   "permissions": [6, 8, 9],
   "users": []
}
```
Create ODBC-only user

An administrator can create a user with the ODBC-only role with the following request. The administrator must have the Administrator (12) permission, or the CreateUsers (13) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/users

Request Payload

```
{
    "userName": "ODBCUser",
    "tenantId": 56,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "password": "TempPassword",
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [
            42
        ]
    }
}
```

Response Payload

```
{
    "id": 963,
    "userName": "ODBCUser",
    "tenantId": 56,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "passwordStatus": 1,
        "passwordExpiration": null
    },
    "permissions": {
        "roles": [
            42
        ]
    },
    "authenticationInfo": {
        "authUsers": [
            {
                "authUserName": "ODBCUser",
                "authServiceId": 1
            }
        ]
    }
}
```
Create a data source on behalf of ODBC-only user

An administrator can create a data source on behalf of ODBCUser with the following request. While the user will not be able to view data source information or modify the data source, ODBCUser will be able to execute ODBC queries on the data source and change their password in the Web UI.

The user query parameter (?user) is used to specify the owner of the data source. The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the CreateDataSource (1) permission.

Request

POST https://MyServer:8443/api/mgmt/datasources?user=ODBCUser

Request Payload

```json
{
   "name": "Oracle_ODBC",
   "dataStore": 43,
   "connectionType": "Hybrid",
   "description": "",
   "options": {
       "User": "OracleTest",
       "Password": "Secret",
       "ServerName": "TestServer",
       "ODataSchemaMap": "{"odata_mapping_v2":{"schemas":[]}}",
       "ODataVersion": "2",
       "ExtendedOptions": "EncryptionMethod=noEncryption",
       "SID": "UNI"
   }
}
```

Response Payload

```json
{
   "id": "2918",
   "name": "Oracle_ODBC",
   "dataStore": 43,
   "connectionType": "Hybrid",
   "description": "",
   "options": {
       "User": "OracleTest",
       "Password": "Secret",
       "ServerName": "TestServer",
       "ODataSchemaMap": "{"odata_mapping_v2":{}},{"schema":[]}
   }
}
```

Update ODBC-only role to include SQL Editor access

With the following request, an administrator can update the ODBC-only role to include SQL editor access. The SQLEditor permission allows the user to pass SQL queries with the SQL Editor in the Web UI. To use the SQL Editor functionality, Web UI permission must also be granted. The administrator must have the Administrator (12) permission, or the ModifyRole (19) permission and administrative access on the tenant.

Note: The payload should also include any previously set permissions that need to be retained, as well as the user or users assigned the role.
Grant SQL Editor access explicitly to the ODBC-only user

Alternatively, an administrator could explicitly set the SQLEditor permission on the user. To use the SQL Editor functionality, Web UI permission must also be granted. In this example, the user inherits ODBC, WebUI, and change password permissions through the ODBC-only Users role (42), while the SQLEditor (10) permission is set explicitly on the user. The administrator must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant to which the user belongs.

**Note:** The request payload must include the roles the user needs to retain. The payload should also include any previously set explicit permissions the user needs to retain.
Managing permissions with Hybrid Data Pipeline APIs

The Hybrid Data Pipeline APIs can be used to manage permissions for a user, role, or data source. The following topics provide a number of example operations for the handling of permissions.

- Retrieving permissions on page 133
- Working with roles on page 136
- Working with user permissions on page 139
- Working with data source permissions on page 141

Retrieving permissions

The first step in working with permissions may simply be retrieving permissions. An administrator may want to retrieve a list of all supported permissions, or retrieve the permissions for a role, user, or data source.

- Retrieve supported permissions
- Retrieve roles and permissions on a role
- Retrieve effective permissions on a user
- Retrieve permissions on a data source

Note: Administrators can also retrieve permissions on data sources that are shared with users and tenants. See Data Sources API on page 1277 and Sharing data sources on page 1279 for details.

Retrieve supported permissions

An administrator can retrieve information on all supported permissions using the Administrator Permissions API. A user must have either the Administrator (12) or MgmtAPI (11) to use this API.

Request

GET https://MyServer:8443/api/admin/permissions

Response Payload

```
{
  "permissions": [
    {
      "id": 1,
      "name": "CreateDataSource",
      "description": "May create new data sources."
    },
    {
      "id": 2,
      "name": "ViewDataSource",
      "description": "May view data sources."
    }
  ]
}
```
Retrieve roles and permissions on a role

A role ID is required to retrieve permissions on a role. Therefore, an administrator may need to retrieve roles before requesting permissions on a role. The Roles API can be used to retrieve roles and then permissions associated with a specific role.

Retrieve roles

The following request retrieves the roles for a Hybrid Data Pipeline service. The user must have the Administrator (12) permission, or the ViewRole (18) permission and administrative access on the tenant.

Request

GET https://MyServer:8443/api/admin/roles

Response Payload

```
{
  "roles": [
    {
      "id": 1,
      "name": "Administrator",
      "tenantId": 1,
      "description": "This role has all permissions. This role cannot be modified or deleted."
    },
    {
      "id": 2,
      "name": "User",
      "tenantId": 1,
      "description": "This role has the default permissions that a normal user will be expected to have."
    },
    {
      "id": 3,
      "name": "Tenant Administrator",
      "tenantId": 1,
      "description": "This role has all the tenant administrator permissions."
    }
  ]
}
```

Retrieve permissions on a role

With the role ID, an administrator can retrieve the permissions associated with a role. This request also returns the users that have been assigned the role. The user must have the Administrator (12) permission, or the ViewRole (18) permission and administrative access on the tenant.

Request

https://MyServer:8443/api/admin/roles/2
Response Payload

```
{
    "name": "User",
    "tenantId": 1,
    "description": "This role has the default permissions that a normal user will be expected to have.",
    "permissions": [
        1,
        2,
        3,
        4,
        5,
        6,
        7,
        8,
        9,
        10,
        11
    ],
    "users": [
        2,
        9,
        46
    ]
}
```

Retrieve effective permissions on a user

An administrator can retrieve permissions on a user with either the Management Permissions API or the Users API. The permissions for a user are the sum of the permissions granted to the user's role(s) and permissions granted explicitly to the user.

Management Permissions API example

The following Management Permissions API request returns the list of effective permissions for the user by specifying the user's name with the user query parameter (?user). The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and administrative access on the tenant to which the user belongs.

Request

```
GET https://MyServer:8443/api/mgmt/permissions?user=d2cuser
```

Response Payload

```
{
    "userId": 2,
    "permissions": [
        1,
        2,
        3,
        4,
        5,
        6,
        7,
        8,
        9,
        10,
        11
    ]
}
```

Users API example

```
Response Payload

```
The following Users API request returns a roles object that shows the roles assigned to the user, and a permissions object that shows the permissions that have been explicitly set on the user. The \{id\} is the auto-generated user ID. The administrator must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant to which the user belongs.

**Request**

GET https://MyServer:8443/api/admin/users/{id}/permissions

**Response Payload**

```json
{
    "roles": [5],
    "permissions": [8, 9, 10]
}
```

**Retrieve permissions on a data source**

The following Data Sources API request retrieves permissions on a specific data source on behalf of the data source owner. The \{datasourceId\} is the auto-generated data source ID, and the user query parameter (\?user) is used to specify the owner of the data source. The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the ViewDataSource (2) permission.

**Note:** When no permissions have been set on a data source, then the permissions of the user are returned. When permissions have been set on a data source, they will be returned instead of the user's permissions. The permissions on a data source override the user's permissions.

**Request**

GET https://MyServer:8443/api/mgmt/datasources/{datasourceId}/permissions?user=TestUser

**Request Payload**

```json
{
    "permissions": [2, 5]
}
```

**Working with roles**

The following operations show how the Roles API can be used to retrieve roles, create roles, retrieve details on a role, and update the permissions on a role.

**Note:** Hybrid Data Pipeline provides three default roles in the system tenant: System Administrator, Tenant Administrator, and User. The System Administrator role has all permissions, the Tenant Administrator role has tenant and user permissions, and the User role has only user permissions. These roles cannot be deleted, and only the users associated with them can be modified. (See also Permissions and default roles.)

- Retrieve current roles
Create a new role

Retrieve details on new role

Update permissions on new role

Retrieve current roles

The following request will retrieve current roles in the Hybrid Data Pipeline service. The administrator must have the Administrator (12) permission, or the ViewRole (18) permission and administrative access on the tenant.

Request

GET https://MyServer:8443/api/admin/roles

Note: The ?tenantID=<tenant_id> and ?tenantName=<tenant_name> query parameters can be appended to the URL to limit the roles returned to a specific tenant.

Response Payload

```
{
   "roles": [
   {
      "id": 1,
      "name": "Administrator",
      "tenantId": 1,
      "description": "This role has all permissions. This role cannot be modified or deleted."
   },
   {
      "id": 2,
      "name": "User",
      "tenantId": 1,
      "description": "This role has the default permissions that a normal user will be expected to have."
   },
   {
      "id": 3,
      "name": "Tenant Administrator",
      "tenantId": 1,
      "description": "This role has all the tenant administrator permissions."
   }
]
```

Create a new role

With the following POST request, a new role is created which allows OData-only access to three users as specified with the "users" property. The administrator must have the Administrator (12) permission, or the CreateRole (17) permission and administrative access on the tenant.

Request

POST https://MyServer:8443/api/admin/roles

Request Payload

```
{
   "name": "ODataOnly",
   "tenantId": 1,
   "description": "This role allows only OData access.",
   "permissions": [7],
```
"users": [11, 12, 13]
}

Response Payload
{
    "id": 37,
    "name": "ODataOnly",
    "tenantId": 1,
    "description": "This role allows only OData access.",
    "permissions": [7],
    "users": [11, 12, 13]
}

Retrieve details on new role

An administrator can then retrieve details on the new role, including permissions and users, with the following GET request. The role ID 37 is past in the request URL. The administrator must have the Administrator (12) permission, or the ViewRole (18) permission and administrative access on the tenant.

Request
GET https://MyServer:8443/api/admin/roles/37

Response Payload
{
    "id": 37,
    "name": "ODataOnly",
    "tenantId": 1,
    "description": "This role allows only OData access.",
    "permissions": [7],
    "users": [11, 12, 13]
}

Update permissions on new role

An administrator can also use a PUT request to update permissions and users associated with the new role. The following request adds the SQLEditor permission to the role and assigns the role to an additional user. The administrator must have the Administrator (12) permission, or the ModifyRole (19) permission and administrative access on the tenant.

Request
PUT https://MyServer:8443/api/admin/roles/37

Request Payload
{
    "id": 37,
    "name": "ODataOnly",
    "tenantId": 1,
"description": "This role allows OData access and access to the Web UI SQL editor.",
"permissions": [
  7,
  10
],
"users": [
  11,
  12,
  13,
  14
]
}

Response Payload
{
  "id": 37,
  "name": "ODataOnly",
  "tenantId": 1,
  "description": "This role allows OData access and access to the Web UI SQL editor.",
  "permissions": [
    7,
    10
  ],
  "users": [
    11,
    12,
    13,
    14
  ]
}

Working with user permissions

Administrators can use the Users API to create users with a specific role and set permissions explicitly on users. The permissions for a user are the sum of the permissions granted to the user's role(s) and permissions granted explicitly to the user. When creating a user, the administrator must assign the user a role.

**Note:** Administrators cannot use the Users API to assign themselves a role or set permissions on themselves. Such tasks would have to be done by another administrator. Best practices recommend that there should be at least two users with Administrator (12) permission. Any user with the Administrator (12) permission is in effect a system administrator and has permission to use all Hybrid Data Pipeline features and functionality.

- **Create a new user**
- **Set explicit permissions on the user**
- **Retrieve permissions on the new user**

Create a new user

The following POST creates a user with the ODataOnly role. The user inherits the permissions associated with this role. The administrator must have the Administrator (12) permission, or the CreateUser (13) permission and administrative access on the tenant.

**Request**

POST https://MyServer:8443/api/admin/users
Set explicit permissions on the user

An administrator can then set permissions explicitly on the new user with the following PUT request, where `{id}` is the auto-generated user ID. In this example, the user is explicitly being granted ChangePassword permission. The administrator must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.

Request

```
PUT https://MyServer:8443/api/admin/users/{id}/permissions
```

Request Payload

```
{
  "roles": [6],
  "permissions": [10]
}
```
Response Payload

```
{
  "roles": [6],
  "permissions": [10]
}
```

Retrieve permissions on the new user

With the following GET request, the permissions in terms of roles and explicit permissions can be retrieved for the new user, where \{id\} is the auto-generated ID of the user. The administrator must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

Request

```
GET https://MyServer:8443/api/admin/users/{id}/permissions
```

Response Payload

```
{
  "roles": [6],
  "permissions": [10]
}
```

Working with data source permissions

The Data Sources API allows administrators to create their own data sources and create data sources on behalf of users. When creating a data source on behalf of a user, administrators can set permissions on the data source to limit user access to the data source. Data source permissions override individual user permissions whether inherited through a role or set explicitly for the user. When an administrator creates a data source on behalf of a user, any administrator with the appropriate permissions would have access to the data source through the on-behalf-of functionality.

- Create a data source on behalf of a user
- Retrieve permissions on behalf of a user
- Update permissions on a data source
- Retrieve the effective permissions on a data source

Create a data source on behalf of a user

The following POST request creates a data source on behalf of a user. The user query parameter (?user) is used to specify the owner of the data source. The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the CreateDataSource (1) permission.

Request

```
```
Request Payload

```json
{
  "name": "ODataSF",
  "dataStore": "1",
  "connectionType": "Cloud",
  "description": "Test OData access to Salesforce",
  "options": {
    "Database": "Accounting",
    "User": "mySForceUserId",
    "Password": "mySForcePassword",
    "SecurityToken": "mySecurityToken",
    "StmtCallLimit": "60",
    "ODataSchemaMap": "\"odata\_mapping\_v2\":\"schemas\":\{"\"name\":\"D2CQA01\"
      ,\"tables\":\{"\"Dept\_Emp\":{},\"Employees\":{},\"Departments\":{},\"Salaries\":{}
      ,\"Titles\":{},\"Dept\_Manager\":{}\}\}},",
    "ODataVersion": "2"
  }
}
```

Retrieve permissions on behalf of a user

The following GET request retrieves the effective permissions on the data source on behalf of the data source owner, where 16 is the ID of the data source.

The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the ViewDataSource (2) permission.

**Note:** When no permissions have been set on a data source, then the permissions of the user are returned. When permissions have been set on a data source, they will be returned instead of the user's permissions. The permissions on a data source override user and role permissions.

Request

```
GET https://MyServer:8443/api/mgmt/datasources/16/permissions?user=ODataUser
```

Response Payload

```json
{
  "permissions": [7]
}
```

Update permissions on a data source

With the following PUT request, an administrator can modify permissions on the data source on behalf of the data source owner. In this example, the administrator allows the ODataUser several additional permissions.

The user query parameter (user) is used to specify the owner of the data source. The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the ModifyDataSource (3) permission.

Request

```
PUT https://MyServer:8443/api/mgmt/datasources/16/permissions?user=ODataUser
```
Request Payload

```
{
    "permissions": [2, 3, 4, 7, 10]
}
```

**Retrieve the effective permissions on a data source**

An administrator can then retrieve the updated effective permissions with a GET request.

The user query parameter `(?user)` is used to specify the owner of the data source. The administrator must have the Administrator (12) permission; or the administrator must have the MgmtAPI (11) permission, the OnBehalfOf (21) permission, administrative access on the tenant to which the user belongs, and the ViewDataSource (2) permission.

**Note:** When permissions have been set on a data source, the effective permissions are the permissions set on the data source. Since data source permissions override user permissions, the user permissions are excluded from the response payload.

**Request**

```
GET https://MyServer:8443/api/mgmt/datasources/16/permissions?user=ODataUser
```

**Response Payload**

```
{
    "permissions": [2, 3, 4, 7, 10]
}
```

**Authentication**

Hybrid Data Pipeline supports internal and external authentication. When the default internal authentication system is used, end user credentials are checked against a hash of the password stored in the Hybrid Data Pipeline account database. When external authentication is used, end user credentials are checked against an external authentication service. External authentication services may be supported either through a Java plugin or through an LDAP server. The following topics provide details and procedures for implementing authentication services.

- Integrating external authentication with a Java plugin on page 144
- Integrating an LDAP authentication service on page 153
- Advanced functionality for authentication services on page 157
Integrating external authentication with a Java plugin

Hybrid Data Pipeline supports external authentication services through a Java authentication plugin. The following general steps must be followed to implement authentication with a Java plugin.

**Note:** If running Hybrid Data Pipeline in FIPS mode, the Java authentication plugin must be FIPS compliant. In addition, the plugin should be tested with FIPS mode enabled before moving to a production environment.

1. Build a Java plugin that implements the Java authentication plugin interface using the `authjavaplugin.jar` file provided in the product package.
2. Add the Java plugin to the Hybrid Data Pipeline environment.
3. Register a Java plugin authentication service.
4. Configure a Hybrid Data Pipeline user account to authenticate end user credentials against the Java plugin authentication service.

Building a Java plugin for external authentication

The first step in integrating a Java authentication plugin is building the plugin. The plugin must be built using Java 7 or 8.

The external authentication service must be multi-thread safe. In other words, Hybrid Data Pipeline must be able to safely have multiple threads call `authenticate()` on the same Java plugin object at the same time. The Hybrid Data Pipeline service must also be able to create multiple instances of the plugin.

Take the following steps to build a Java plugin to use with an external authentication service.

1. Create a Java class that implements the Java authentication plugin interface, according to substeps a, b, and c.

   The Java authentication plugin interface is:
   ```java
   com.ddtek.cloudservice.plugins.auth.javaplugin.JavaAuthPluginInterface
   ```

   The Java authentication plugin interface is defined in the `<install_dir>/ddcloud/dev/lib/authjavaplugin.jar`, where `<install_dir>` is the installation directory of a Hybrid Data Pipeline server.

   See [Java authentication plugin interface syntax](#) on page 145 for the syntax of the interface definition.

   See [Java authentication plugin sample](#) on page 146 for an example plugin.

   a) After creating an instance of the Java plugin, Hybrid Data Pipeline will call the `init()` method in the object to initialize the object with configuration information.

   ```java
   void init(HashMap<String, Object> attributes, Logger logger)
   ```

   **attributes:** a JSON object that can provide useful values for initialization, such as an authentication server name. Multiple authentication services can use the same plugin as long as the appropriate attributes are provided via the JSON object. Hybrid Data Pipeline passes a HashMap representation of the JSON object for any authentication service configured to use the plugin and registered via the Authentication API.
logger: an object that can be used to log information, such as failed authentication or errors that occurred when authenticating a user. The log entries are collected in a separate file named extauth<date>.log located in the .../ddcloud/das/server/logs/das subdirectory.

b) The following method is called by the Hybrid Data Pipeline service to release or close resources in the event Hybrid Data Pipeline shuts down or the authentication service is updated.

```java
void destroy()
```

c) The Hybrid Data Pipeline service calls the following method to authenticate the Hybrid Data Pipeline end user.

```java
boolean authenticate(String username, String password, String ipAddress)
```

- **username**: the username persisted by an authentication service. Referred to as the authUserName in the Users API.
- **password**: the password provided by the end user.
- **ipAddress**: the IP Address of the end user machine.

**Note:** If the user cannot be authenticated, an error is returned. When the plugin returns false, Hybrid Data Pipeline will return an invalid username and password error. If the plugin throws an exception, Hybrid Data Pipeline will return an error indicating the service is unavailable.

2. Compile the Java class implemented in Step 1 with any other Java classes needed to implement the authentication methods.

   The following command compiles the Java class.

   ```bash
   javac -cp <install_dir>/ddcloud/dev/lib/authjavaplugin.jar ...
   ```

3. Package all the class files into a jar file.

   The following command packages input files into the file custom_auth_plugin.jar.

   ```bash
   jar cf custom_auth_plugin.jar <inputs>
   ```

**What to do next:**

The Java authentication plugin, in the form of the jar file, must be added to the Hybrid Data Pipeline environment.

**Java authentication plugin interface syntax**

When building a Java plugin, a Java class must be created that implements the Java authentication plugin interface. The Java plugin interface has the following syntax.

```json
{
   "className": "java_plugin_classname",
   "attributes": {
      "attribute_name": "attribute_value",
      "attribute_name": "attribute_value",
      ...
   }
}
```
Valid Values Description Property

"className" The class name that implements the Java authentication plugin interface. The name of the class that the Java plugin developer created to implement the Java authentication plugin interface.

"attributes" A JSON object comprised of named attribute values that are passed to the init method of the Java plugin. These attributes can provide useful values for initialization, such as an authentication server name, and can be used to configure the plugin for use by multiple authentication servers. A valid JSON object.

Interface example

```json
{
    "className": "com.test.hdp.plugins.auth.HDPUserAuthentication",
    "attributes": {
        "Server": "test-authentication",
        "BackupServer": "test-authentication-backup"
    }
}
```

Java authentication plugin sample

The following sample Java authentication plugin can be modified to create a custom plugin for integrating an in-house authentication system with Hybrid Data Pipeline. The external authentication service must be multi-thread safe. In other words, Hybrid Data Pipeline must be able to safely have multiple threads call authenticate() on the same Java plugin object at the same time. The Hybrid Data Pipeline service must also be able to create multiple instances of the plugin.

The Java authentication plugin interface must be implemented with the following methods.

- void init (HashMap<String, Object> attributes, Logger logger)
- void destroy ()
- boolean authenticate (String username, String password, String ipAddress)

The `JavaAuthPluginException` constructor can be used to handle errors and exceptions.

```java
package com.ddtek.cloudservice.plugins.auth.javaplugin;

import java.util.HashMap;
import java.util.Iterator;
import java.util.Properties;
import java.util.Set;
import java.util.logging.Logger;

public class JavaAuthPluginSample implements JavaAuthPluginInterface {

    Properties authorizedUsers;

    /**
     * Initializes a Java authentication plugin with any properties specified when defining the plugin.
     * @param props The defined properties for this plugin.
     * @param logger A Java logger for the plugin to use.
     */

    public JavaAuthPluginSample (Properties props, Logger logger) {
        this.authorizedUsers = props;
    }

    @Override
    public void init (HashMap<String, Object> attributes, Logger logger) {
        // Initialize the plugin
    }

    @Override
    public void destroy () {
        // Clean up
    }

    @Override
    public boolean authenticate (String username, String password, String ipAddress) {
        // Authenticate
        return true;
    }

    public void getAllUsers () {
        // Get all users
    }
}
```
@Override
public void init (HashMap<String, Object> attributes, Logger logger) {
    if (attributes == null) {
        authorizedUsers = new Properties ();
        authorizedUsers.setProperty ("d2ctest", "d2ctest");
        return;
    }
    authorizedUsers = new Properties ();
    Set<String> keySet = attributes.keySet ();
    Iterator<String> keys = keySet.iterator ();
    while (keys.hasNext ()) {
        String key = keys.next ();
        Object value = attributes.get (key);
        if (value instanceof String) {
            authorizedUsers.setProperty (key, (String) value);
        } else {
            logger.warning (value.toString () + " [" + value.getClass ().getName () + "] is not a String");
        }
    }
}
/**
 * Terminates a Java authentication plugin -- free resources and cleanup.
 */
@Override
public void destroy () {}
/**
 * Authenticates a username and password.
 * If authentication cannot be determined, such as due to a failure
 * in the authentication mechanism, an exception should be thrown.
 * This routine must be multi-thread safe.
 * @param username The name of the user.
 * @param password The password to authenticate with.
 * @param ipAddress The IP address of the authentication request.
 * @returns Whether or not the username and password are valid.
 */
@Override
public boolean authenticate (String username, String password, String ipAddress) {
    String pwd = authorizedUsers.getProperty (username);
    // Assumes password is never null, but pwd may be null.
    return password.equals (pwd);
}
/**
 * Constructor for JavaAuthPluginException.
 */
public JavaAuthPluginException ();
/**
 * Constructor for JavaAuthPluginException.
 * @param message Detail message for JavaAuthPluginException.
 */
public JavaAuthPluginException (String message);
/**
 * Constructor for JavaAuthPluginException.
 * @param message Detail message for JavaAuthPluginException.
 * @param cause Cause of the exception.
 */
Adding a Java authentication plugin to a Hybrid Data Pipeline environment

Once the Java authentication plugin has been built as described in Building a Java plugin for external authentication on page 144, the plugin must be added to the Hybrid Data Pipeline environment.

Take the following steps to add a Java authentication plugin.

1. Add the plugin and any other jar files required for the implementation, such as Apache HTTP Client jars, to the plugins directory. The location of the plugins directory depends on the Hybrid Data Pipeline deployment.

   Standalone node deployment
   The plugins directory will be found in either of the following locations.
   • <install_dir>/ddcloud/keystore if the default key location was selected during installation of the server
   • user_specified_location if a non-default key location was specified during installation of the server

   Load balancer deployment
   The plugins directory will be found in the location specified as the key location during installation of the server.

2. Restart the Hybrid Data Pipeline service on each node that is running the service.
   a) Run the stop service script for each node running the service. The location of the stop script is install_dir/ddcloud/stop.sh.

      Note: Shutting down Hybrid Data Pipeline can take up to 2.5 minutes. Wait until you see the Shutdown complete message displayed on the console before taking any additional actions.

   b) Run the start service script for each node running the service. The location of the start script is install_dir/ddcloud/start.sh.

What to do next:
The external authentication service must be registered using the Authentication API.

Registering a Java plugin authentication service

Before a user account can be configured to use a Java plugin authentication service, the authentication service must be registered in Hybrid Data Pipeline. As described in the following sections, you can register a Java plugin authentication service either through the Web UI or the Authentication API.

Note:
• An external authentication service registered in the default system tenant is available across all tenants, while an external authentication service registered in a child tenant is only available in that tenant. Once a
service is registered with a tenant, the tenant administrator can create or modify user accounts to authenticate end user credentials against the service.

- A user with the Administrator (12) permission can register an external authentication service on any tenant within the system. A user with the RegisterExternalAuthService (26) permission can register an external authentication service on any tenant to which he or she has administrative access.

**Register Java plugin service via the Web UI**

Take the following steps to register a Java plugin service via the Web UI.

1. Navigate to the Manage External Authentication view by clicking the manage external authentication icon.

2. Select the tenant for which you are registering the service from the Select Tenant dropdown.

3. Click + New Service. You will be directed to the Create Authentication Service screen.

4. Provide the following information.
   - The name and description of the service
   - The service type
   - The class name (The class name that implements the Java authentication plugin. For example, com.sample.plugin.auth.JavaPluginAuthSample.)
   - Attributes (A JSON object comprised of named attribute values that are passed to the init method of the Java plugin.)

5. Click Save.

**What to do next:**

Configure Hybrid Data Pipeline user accounts to authenticate end user credentials against the Java plugin authentication service. See Configuring user accounts for Java plugin authentication on page 150 for details.

**Register Java plugin service via the Authentication API**

The following POST operation registers the jplugauth service. The className property provides the class name of the Java plugin, and the attributes property provides the HashMap that will be processed by the authentication service. For further details, see Register an external authentication service.

**Request**

```
POST https://MyServer:8443/api/admin/auth/services
```

**Request payload**

```
{
   "name": "jplugauth",
   "tenantId": 1,
   "description": "Java external auth plugin",
   "authDefinition": {
      "className": "com.test.hdp.plugins.auth.HDPUserAuthentication",
      "attributes": {
         "Server": "test-authentication",
         "BackupServer": "test-authentication-backup"
      }
   },
   "authTypeId": 2
}
```
Response payload

Status code: 201
Successful response

{
   "id": 43,
   "name": "jplugauth",
   "tenantId": 1,
   "description": "Java external auth plugin",
   "authDefinition": {
      "className": "com.test.hdp.plugins.auth.HDPUserAuthentication",
      "attributes": {
         "Server": "test-authentication",
         "BackupServer": "test-authentication-backup"
      }
   }
   "lastModifiedTime": "2018-02-15T11:09:35.107Z",
   "authTypeId": 2,
   "tenantName": "OrgM"
}

What to do next:
Configure Hybrid Data Pipeline user accounts to authenticate end user credentials against the Java plugin authentication service. See Configuring user accounts for Java plugin authentication on page 150 for details.

Configuring user accounts for Java plugin authentication

Once a Java plugin service has been registered, user accounts can be configured to use the service. As described in the following sections, user accounts can be configured through either the Web UI or the Users API.

Using the Web UI to configure a user account for Java plugin authentication

To create a new user account, take the following steps.

1. Navigate to the Manage Users view by clicking the manage users icon.
2. Click + New User.
3. Under the General tab, provide tenant, user name, and user role information.
4. Click the Authentication Setup tab.
   - Option 1. If you are adding the Java plugin service as an additional authentication type for the user account, click + Add Authentication Service.
   - Option 2. If you want to use only the Java plugin service, modify the properties of the current authentication type.
5. Select the Java plugin service from the Authentication Type dropdown.
6. In the External Usernames field, specify the user or users you want to associate with the Hybrid Data Pipeline user account. Any user name provided should correspond to a user name persisted by the authentication service.
7. Click Save.

To modify a current user account, take the following steps.
1. Navigate to the **Manage Users** view by clicking the manage users icon.

2. From the list of user accounts, click the user account you want to modify.

3. Click the **Authentication Setup** tab.

   - **Option 1.** If you are adding the Java plugin service as an additional authentication type for the user account, click **+ Add Authentication Service**.
   - **Option 2.** If you want to use only the Java plugin service, modify the properties of the current authentication type.

4. Select the Java plugin service from the **Authentication Type** dropdown.

5. In the **External Usernames** field, specify the user or users you want to associate with the Hybrid Data Pipeline user account. Any username provided should correspond to a user name persisted by the authentication service.

6. Click **Update** to save your changes to the user account.

### Using the Users API to configure a user account for Java plugin authentication

To create a new user, take the following steps.

The following POST operation creates a user account using an external authentication service. Here the end user (**user_external**) authenticates via a Java plugin external authentication service (**authServiceId**: 43). This end user inherits all the attributes associated with the **testuser** account. For further details, see [Create a user account](#).

**Request**

```plaintext
POST https://MyServer:8443/api/admin/users
```

**Request payload**

```json
{

"userName": "testuser",

"tenantId": 1,

"statusInfo": {

  "status": 1,

  "accountLocked": false

},

"passwordInfo": {

  "passwordStatus": 1,

  "passwordExpiration": "2020-01-01 00:00:00"

},

"permissions": {

  "roles": [2]

},

"authenticationInfo": {

  "authUsers": [

    {

      "authUserName": "user_external",

      "authServiceId": 43

    }

  ]

}

}
```

---

To modify a current user account, take the following steps.

The following PUT operation updates user account 101 to use the a Java plugin external authentication service ("authServiceId": 43) for managing authentication. Two end users (user_1 and user_2) have been associated with the account. Their credentials are managed through the authentication service that has ID 43. Each user inherits all the attributes associated with user account 101. For further details, see Update authentication information on a user account.

Request

PUT https://MyServer:8443/api/admin/users/101/authinfo

Request payload

```
{
  "authUsers": [
    {
      "authUserName": "user_1",
      "authServiceId": 43
    },
    {
      "authUserName": "user_2",
      "authServiceId": 43
    }
  ]
}
```

Response payload

```
Status code: 200
Successful response
```
Integrating an LDAP authentication service

LDAP authentication services can be integrated with Hybrid Data Pipeline. The following general steps apply to integrating an LDAP service.

1. The LDAP service must be registered as an external authentication service.
2. Hybrid Data Pipeline user accounts must be configured to use the LDAP service.

Registering an LDAP authentication service

Before a user account can be configured to use LDAP, an LDAP service must be registered with Hybrid Data Pipeline. As described in the following sections, you can register a Java plugin authentication service either through the Web UI or the Authentication API.

Note:

- An external authentication service registered in the default system tenant is available across all tenants, while an external authentication service registered in a child tenant is only available in that tenant. Once a service is registered with a tenant, the tenant administrator can create or modify user accounts to authenticate end user credentials against the service.

- A user with the Administrator (12) permission can register an external authentication service on any tenant within the system. A user with the RegisterExternalAuthService (26) permission can register an external authentication service on any tenant to which he or she has administrative access.

Register LDAP service via the Web UI

Take the following steps to register an LDAP service via the Web UI.

1. Navigate to the Manage External Authentication view by clicking the manage external authentication icon.
2. Select the tenant for which you are registering the service from the Select Tenant dropdown.
3. Click + New Service. You will be directed to the Create Authentication Service screen.
4. Provide the following information.
   - The name and description of the service
   - The service type
   - Target URL (The URL used to access the LDAP service.)
   - Service Authentication (The authentication mechanism required by the LDAP service.)
• Security Principal (The principal used to authenticate against the LDAP server. The user name token %LOGINNAME% is supported to permit the replacement of the actual user name. For example, CN=%LOGINNAME%,OU=TestRuns,DC=testdomain.)

• Other Attributes (A valid JSON Object to be passed as key and value pairs to the environment properties during the creation of InitialDirContext object.)

5. Click Save.

What to do next:
Configure Hybrid Data Pipeline user accounts to use the LDAP service. See Configuring user accounts for LDAP authentication on page 155 for details.

Register LDAP service via the Authentication API
The following POST operation registers the LDAP service. For further details, see Register an external authentication service.

Request

POST https://MyServer:8443/api/admin/auth/services

Request payload

{  "name": "LDAP1",  "tenantId": 1,  "description": "LDAP Auth plugin",  "authDefinition": {    "attributes": {      "targetUrl": "LDAP://123.45.67.899:389",      "securityAuthentication": "simple",      "securityPrincipal": "CN=%LOGINNAME%,OU=TestRuns,DC=testdomain,DC=local"    }  },  "authTypeId": 3}

Response payload

Status code: 201
Successful response

{  "id": 21,  "name": "LDAP1",  "tenantId": 1,  "description": "LDAP Auth plugin",  "authDefinition": {    "attributes": {      "targetUrl": "LDAP://123.45.67.899:389",      "securityAuthentication": "simple",      "securityPrincipal": "CN=%LOGINNAME%,OU=TestRuns,DC=testdomain,DC=local"    }  },  "lastModifiedTime": "2018-02-14T11:34:13.009Z",  "authTypeId": 3,  "tenantName": "OrgT"}
What to do next
Configure Hybrid Data Pipeline user accounts to use the LDAP service. See Configuring user accounts for LDAP authentication on page 155 for details.

Configuring user accounts for LDAP authentication
Once an LDAP service has been registered, user accounts can be configured to use the service. As described in the following sections, user accounts can be configured through either the Web UI or the Users API.

Using the Web UI to configure a user account for LDAP authentication
To create a new user account, take the following steps.

1. Navigate to the Manage Users view by clicking the manage users icon.
2. Click + New User.
3. Under the General tab, provide tenant, user name, and user role information.
4. Click the Authentication Setup tab.
   - Option 1. If you are adding the LDAP service as an additional authentication type for the user account, click + Add Authentication Service.
   - Option 2. If you want to use only the LDAP service, modify the properties of the current authentication type.
5. Select the LDAP service from the Authentication Type dropdown.
6. In the External Usernames field, specify the user or users you want to associate with the Hybrid Data Pipeline user account. Any username provided should correspond to a username persisted by the authentication service.
7. Click Save.

To modify a current user account, take the following steps.

1. Navigate to the Manage Users view by clicking the manage users icon.
2. From the list of user accounts, click the user account you want to modify.
3. Click the Authentication Setup tab.
   - Option 1. If you are adding the LDAP service as an additional authentication type for the user account, click + Add Authentication Service.
   - Option 2. If you want to use only the LDAP service, modify the properties of the current authentication type.
4. Select the LDAP service from the Authentication Type dropdown.
5. In the External Usernames field, specify the user or users you want to associate with the Hybrid Data Pipeline user account. Any username provided should correspond to a user name persisted by the authentication service.
6. Click Update to save your changes to the user account.

Using the Users API to configure a user account for LDAP authentication
To create a new user account, take the following steps.
The following POST operation creates a user account that authenticates through an LDAP service. Here the end user (LDAP_user_1) authenticates via an LDAP service ("authServiceId": 21). This end user inherits all the attributes associated with the testuser2 account. For further details, see Create a user account.

**Request**

```
POST https://MyServer:8443/api/admin/users
```

**Request payload**

```json
{
    "userName": "testuser2",
    "tenantId": 1,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "passwordStatus": 1,
        "passwordExpiration": "2020-01-01 00:00:00"
    },
    "permissions": {
        "roles": [2]
    },
    "authenticationInfo": {
        "authUsers": [{
            "authUserName": "LDAP_user_1",
            "authServiceId": 21
        }]
    }
}
```

**Response payload**

```json
{
    "id": 8,
    "userName": "testuser2",
    "tenantId": 1,
    "statusInfo": {
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "passwordStatus": 1,
        "passwordExpiration": "2020-01-01 00:00:00"
    },
    "permissions": {
        "roles": [2]
    },
    "authenticationInfo": {
        "authUsers": [{
            "authUserName": "LDAP_user1",
            "authServiceId": 21
        }]
    }
}
```
To modify a current user account, take the following steps.

The following PUT operation updates user account 202 to use the LDAP service ("authServiceId": 21) for managing authentication. Two end users (user_1 and user_2) have been associated with the account. Their credentials are managed through the authentication service that has ID 21. Each user inherits all the attributes associated with user account 202. For further details, see Update authentication information on a user account.

Request

PUT https://MyServer:8443/api/admin/users/101/authinfo

Request payload

{
   "authUsers": [
   {
      "authUserName": "user_1",
      "authServiceId": 21
   },
   {
      "authUserName": "user_2",
      "authServiceId": 21
   }
   ]
}

Response payload

   Status code: 200
   Successful response

{
   "authUsers": [
   {
      "authUserName": "user_1",
      "authServiceId": 21
   },
   {
      "authUserName": "user_2",
      "authServiceId": 21
   }
   ]
}

Advanced functionality for authentication services

Hybrid Data Pipeline supports the following advanced authentication functionality.

- Integrate multiple authentication services with a single user account
- Associate a group of users to a Hybrid Data Pipeline account using a wildcard
- Set a delimiter for the username credential
Integrate multiple authentication services with a single user account

Multiple authentication services can be integrated with a single Hybrid Data Pipeline user account. After the authentication services have been registered, administrators can configure a user account to use the registered services. In the following API request, an administrator associates a number of end users with a user account named *odata_users* with ID of 18. The *internal_user* uses the internal authentication mechanism. The other end users use separate authentication services as specified with the *authServiceID* property.

Note: You can also associate multiple services (and end users) with a user account through the Web UI. When creating or updating a user account, you can associate an external service with the account by clicking **Add Authentication Service** under the **Authentication Setup** tab.

```json
PUT https://MyServer:8443/api/admin/users/18/authinfo

{  
  "authUsers": [  
    {  
      "authUserName": "internal_user",  
      "authServiceId": 1  
    },  
    {  
      "authUserName": "odata_user_1",  
      "authServiceId": 21  
    },  
    {  
      "authUserName": "odata_user_2",  
      "authServiceId": 43  
    },  
    {  
      "authUserName": "odata_user_3",  
      "authServiceId": 89  
    }  
  ]
}
```

**Associate a group of users to a Hybrid Data Pipeline account using a wildcard**

A wildcard can be used to associate a group of end users in an external authentication service with a user account. The only supported wildcard is *, which matches any and all names. In the following example, an administrator creates a user account called *support_team* and uses a wildcard to associate users in an external authentication service with this account.

**Important:** When a wildcard is used to associate end users with a user account, the Systems Configuration API must be used to implement a delimiter for the username credential as described in the next section.

```json
POST https://MyServer:8443/api/admin/users

{  
  "userName": "support_team",  
  "statusInfo": {  
    "status": 1,  
    "accountLocked": false  
  },  
  "passwordInfo": {  
    "passwordStatus": 1,  
    "passwordExpiration": "2020-01-01 00:00:00"  
  },  
  "permissions": {  
    "roles": [  
      1  
    ]  
  }
}
```
Set a delimiter for the username credential

A delimiter can be specified to require the inclusion of the name of the authentication service, as well as the name of the end user when passing the username credential. A delimiter must be used whenever the wildcard is used to associate names from an external authentication service with a user account. A delimiter should also be required if there is a possibility of naming conflicts among end users from different external authentication services. In the following example, an administrator uses the Systems Configuration API to specify a delimiter.

Note: You can also set a delimiter from the System Configurations view using the Web UI. See System Configurations view on page 79 for details.

```
PUT https://MyServer:8443/api/admin/configurations/1
{
   "value": ":"
}
```

With this implementation, the username credential must take the form auth_user_name:auth_service_name (for example, user437:LDAP1).

Password policy

After installation Hybrid Data Pipeline enforces the following password policy by default.

Note: You must specify passwords for the default d2cadmin and d2cuser accounts during installation of the Hybrid Data Pipeline server. When initially logging in to the Web UI or using the API, you must authenticate as one of these users. Best practices recommend that the passwords adhere to this password policy.

- The password must contain at least 8 characters.
- The password must not contain more than 12 characters. A password with a length of 12 characters is acceptable.
- The password should not contain the username.
- Characters from at least three of the following four groups must be used in the password:
  - Uppercase letters A-Z
  - Lowercase letters a-z
Enabling and disabling the password policy

Hybrid Data Pipeline enforces a password policy by default. When the password policy is turned on, user passwords must conform to the Password policy on page 159.

You can use the Web UI or the System Configurations API to enable or disable the password policy.

Using the Web UI

Take the following steps to enable or disable the password policy via the Web UI.

1. Navigate to the System Configurations view by clicking the system configurations icon.
2. Set Password Policy to the desired value.
3. Click Save.

Using the System Configurations API

The following GET operation retrieves the current behavior. The number 6 is the ID of the password policy attribute.

GET https://MyServer:8443/api/admin/configurations/6

```
{
    "id": 6,
    "description": "Valid values are: 1 or -1. Value of 1 enforces that the password be in compliant with the default password policy. Value of -1 turns off the Password Policy enforcement. Any other value will be treated like -1",
    "value": "-1"
}
```

To disable the default password policy, execute a PUT operation on the same endpoint with the following payload.

```
{
    "value": "-1"
}
```

To enable the default password policy, execute a PUT operation on the same endpoint with the following payload.

```
{
    "value": "1"
}
```

See also

Password policy on page 159
System Configurations view on page 79
System Configurations API on page 1124
Configuring change password behavior

Hybrid Data Pipeline supports two types of change password behavior. By default, change password behavior is configured to require users to provide a current password as well as a new password when changing passwords. Alternatively, change password behavior can be configured such that users are only required to provide and confirm a new password when changing passwords.

You can use the Web UI or the System Configurations API to configure change password behavior.

Using the Web UI

Take the following steps to enable or disable the password policy via the Web UI.

1. Navigate to the System Configurations view by clicking the system configurations icon.
2. Set Secure Password Change.
   - When set to ON, the user must provide a current password before providing new password.
   - When set to OFF, the user need only provide a new password.
3. Click Save.

Using the System Configurations API

Administrators can change the behavior by setting the secureChangePassword attribute in the System Configurations API.

The following PUT operation would configure the system to use the non-default behavior where the user must provide only a new password. The number 2 is the ID of the secureChangePassword attribute.

```
PUT https://<myserver>:<port>/api/admin/configurations/2

{
    "value": "false"
}
```

See also

System Configurations view on page 79
System Configurations API on page 1124

Implementing an account lockout policy

Hybrid Data Pipeline supports the implementation of an account lockout policy. An account lockout policy can be used to limit the number of consecutive failed authentication attempts permitted before a user account is locked. The user is unable to authenticate until a configurable period of time has passed or until the administrator unlocks the account.

The Hybrid Data Pipeline account lockout policy is by default enabled in accordance with Federal Risk and Authorization Management Program (FedRAMP) low- and medium-risk guidelines. The number of failed authentication attempts is limited to 3 in a 15 minute period. Once this limit is met, a lockout of the user account occurs for 30 minutes.
An account lockout policy can only be applied to user accounts managed through the default internal authentication service. A policy cannot be applied to end users managed through an external authentication service.

An account lockout policy can only be applied at the system level. It cannot be applied to individual tenants. To implement an account lockout policy, the administrator must reside in the default system tenant and have either the Administrator (12) or the Limits (27) permission. To unlock a user account, the administrator must have either the Administrator (12) permission or the ModifyUsers (15) permission with administrative access to the tenant. In addition, to use the Web UI for these tasks, the administrator must have either the Administrator (12) or the Web UI (8) permission.

Configuring an account lockout policy

An account lockout policy can be configured either through the Web UI or the Limits API. The following limits are used to define the account lockout policy.

- PasswordLockoutLimit is the number of consecutive failed authentication attempts that are allowed before locking the user account. By default, account lockout functionality is enabled with PasswordLockoutLimit set to 3. Setting PasswordLockoutLimit to zero disables lockout functionality.
- PasswordLockoutInterval is the duration, in seconds, for counting the number of consecutive failed authentication attempts.
- PasswordLockoutPeriod is the duration, in seconds, for which a user account will not be allowed to authenticate to the system when the PasswordLockoutLimit is reached.

Using the Web UI

Take the following steps to configure these limits via the Web UI.

1. Navigate to the Manage Limits view by clicking the manage limits icon.
2. Select the system tenant from the Tenant dropdown.
3. Expand the Security and Password sections to view account policy limits.
4. Specify values for each limit.
5. Click Save.

Using the Limits API

The following PUT operation updates the PasswordLockoutLimit to 5 login attempts. The endpoint is specified with the number 3, the ID of the PasswordLockoutLimit. (See the Limits API on page 1071 for details on setting other account policy lockout limits.)

```
PUT https://myserver:port/api/admin/limits/system/3

{  
    "value": 5
}
```
Unlocking a user account

An account can be unlocked by executing a PUT operation on the statusinfo endpoint in the Users API on page 1146. As the following example shows, the URL must include the user ID, and the payload must include the accountLocked property with a value of false.

```
PUT https://<myserver>:<port>/api/admin/users/{id}/statusinfo

{
    "accountLocked": false
}
```

AccountLockedAt and AccountLockedUntil are additional properties that can be set when unlocking a user account. See Update status info on a user account on page 1167 for further details.

Transactions

Hybrid Data Pipeline supports transactions against data stores that provide transaction support such as DB2, MySQL, Oracle, and SQL Server. Transactions are supported for JDBC, ODBC, and OData client applications. For JDBC and ODBC applications, transactions are handled via the TransactionMode property and Transaction Mode option, respectively. For OData client applications, Hybrid Data Pipeline supports transactions for OData Version 4 batch requests.

Most ODBC and JDBC drivers that support transactions connect to backend data stores using a socket. However, the Hybrid Data Pipeline drivers connect to data stores through an HTTP(S) connection. Therefore, Hybrid Data Pipeline can only detect the abnormal termination of a transaction from a lack of activity on the session. To detect session inactivity, Hybrid Data Pipeline runs a transaction timeout thread through sessions every 5 seconds to look for idle transaction threads. If a transaction remains idle longer than a specified period, it will be rolled back and canceled. By default, the transaction timeout limit is 60 seconds. An administrator can specify the period a transaction thread can remain idle with the transaction timeout limit. The transaction timeout limit can be set at the system, tenant, user, or data source level, or a combination of these. See Manage Limits view on page 76 and Limits API on page 1071 for information on setting limits.

In the following example, an administrator sets the transaction timeout limit to 10 seconds at the system level by executing a POST operation with the Limits API. Given the 5 second interval at which the transaction timeout thread runs, no transaction threads may remain idle for more than 15 seconds with this setting.

Sample request

```
POST https://<myserver>:<port>/api/admin/limits/system/14

{
    "value": 10
}
```

In addition to a transaction timeout, server and session timeouts can also lead to transaction rollback and cancellation. Hybrid Data Pipeline will return the same error for each of these timeouts. When a transaction timeout error is thrown, the connection associated with the error is placed in a special state. The rollback and close methods are allowed for JDBC connections in this state, while only the rollback method is allowed for ODBC connections. However, calls that do not require a round trip to the server may still succeed.

The following isolation levels are supported depending on which isolation levels are supported by the data store.

- TRANSACTION_NONE
- TRANSACTION_READ_UNCOMMITTED
Implementing IP address whitelists

Hybrid Data Pipeline supports IP address whitelists. Using the Hybrid Data Pipeline APIs, administrators can restrict access to Hybrid Data Pipeline by creating a whitelist of IP addresses (either individual IP addresses or a range of IP addresses). IP address whitelists restrict access to Hybrid Data Pipeline resources. In addition, IP address whitelists are implemented at either the system, tenant, or user levels.

**Note:** In the event that an IP address whitelist implementation inadvertently prevents administrators from using Hybrid Data Pipeline, an administrator can bypass the whitelist by accessing the service directly from any machine hosting the service. First, the administrator must have access privileges to the host machine. Then, the administrator can access the service from a host machine by replacing the `servername` in the Hybrid Data Pipeline URL with `localhost`, `127.0.0.1`, or `::1`. Then, the administrator can disable the IP address whitelist or update the implementation as desired.

IP address whitelists restrict access to Hybrid Data Pipeline resources. For example, a system administrator may wish to restrict tenant administrator access to a range of IP addresses for a given tenant. Note that if no whitelist is defined for a particular resource, all IP addresses will be allowed access to that resource. Also, no whitelist restrictions apply when Hybrid Data Pipeline is accessed from a local host. Access to the following resources can be managed with IP address whitelists.

- Management API
- Administrators API
- Data access (ODBC, JDBC, and OData)
- Web UI (system level only)
IP address whitelists are implemented at either the system, tenant, or user levels. When an IP address whitelist is set at the system level, users across the system must access the given Hybrid Data Pipeline resource from an IP address or range of IP addresses specified in the whitelist. When an IP address whitelist is set at the tenant level, users who reside in the tenant must access the resource from IP address or range of IP addresses specified in the whitelist. When an IP address whitelist is set at the user level, the specified user must access the resource from an IP address or range of IP addresses specified in the whitelist. When an IP address whitelist is set at multiple levels for a given resource, Hybrid Data Pipeline first checks the system level, then the tenant level, and then the user level. If any check fails, the user attempting to access the service from an invalid IP address will be unable to log in and a 403 access-denied error will be returned.

Depending on the level at which IP address whitelists are being implemented, an administrator must have certain permissions.

- A administrator with the Administrator (12) permission can implement and create whitelists for all resources at the system, tenant, and user levels.
- A administrator with the following permissions can create whitelists for resources at the tenant level: the MgmtAPI (11) permission, the IPWhiteList (29) permission, and administrative access to the tenant.
- A administrator with the following permissions can create whitelists for resources at the user level: the MgmtAPI (11) permission, the IPWhiteList (29) permission, and administrative access to the tenant to which the user belongs.

### Configuring IP address whitelist

You must use the IP Address Whitelist API to view and configure IP address whitelists. When setting up IP address whitelists, you must identify the IP addresses that you need to whitelist. You can specify a single address, a list of addresses, or a range of addresses.

**Note:** IP address whitelists are enabled by default. Unless you have disabled this feature, any IP address whitelist you create will immediately be enforced. See Enabling and disabling the IP address whitelist feature on page 168 for details.

The IP addresses can be specified in either IPv4 or IPv6 format, or a combination of the two. The IP addresses can also be specified in the IPv4-mapped IPv6 combination address format. The following is the payload format for specifying a range of IP addresses.

```json
{
   "startAddress": "<Starting IP address in IPv4 or IPv6 format>",
   "endAddress": "<Ending IP address in IPv4 or IPv6 format>"
}
```

If you specify only a start address, and do not specify an end address, the specified IP address will be treated as an individual IP address.

If you are specifying a range of IP addresses, the starting IP address and the ending IP address should be in the same format. However, you can specify different IP address formats for different whitelists. For example, you may use the IPv4 format to whitelist data access APIs, but use the IPv6 format to whitelist Management API.

If the incoming IP address is in IPv6 format, it will be validated against the IP address range having IPv6 addresses. This same limitations holds true for IPv4 addresses. The system will not convert IP addresses from one format to another to check for whitelisting.

In a load balanced deployment, the load balancer should be configured to echo back the originating client's IP address in the X-Forwarded-For header to have this feature function appropriately.

In the following section, examples of IP address whitelisting at the system level, tenant level and user level are shown.
**System level example**

In the following example, a GET request retrieves all the IP address whitelists applied at the system level.

**Request**

```
GET https://MyServer:8443/api/admin/security/whitelist/system
```

**Response**

```
{
   "managementAPI": [],
   "adminAPI": [],
   "dataAccess": [],
   "webUI": []
}
```

The response indicates that none of the resources are protected at a system level. The following POST request creates whitelists for all resources except the Web UI. By providing null as the value for the `webUI` property, a whitelist is not applied to the Web UI.

**Request**

```
POST https://MyServer:8443/api/admin/security/whitelist/system
```

**Request Payload**

```
{
   "managementAPI": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.30.10"
      }
   ],
   "adminAPI": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.40.20"
      }
   ],
   "dataAccess": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.50.20"
      }
   ],
   "webUI": null
}
```

If the above request is successful, the specified IP addresses are whitelisted for the specified resources. So, a user with an IP of 10.20.30.5 can access the Management API.

**Tenant level example**

In a multitenant environment, whitelists can be set at a tenant level. In the following example, the POST request creates a whitelist for a specific tenant.

**Request**

```
POST https://MyServer:8443/api/admin/security/whitelist/tenants/2
```

**Request Payload**

```
{
   "managementAPI": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.30.10"
      }
   ],
   "adminAPI": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.40.20"
      }
   ],
   "dataAccess": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.50.20"
      }
   ],
   "webUI": null
}
```
"startAddress": "10.20.30.0",
"endAddress": "10.20.30.5"
},
"adminAPI": [
{
"startAddress": "10.20.30.0",
"endAddress": "10.20.40.5"
}
],
"dataAccess": [
{
"startAddress": "10.20.30.0",
"endAddress": "10.20.50.5"
}
],
"webUI": null
}

If the above request is successful, the specified IP addresses are whitelisted for the specified resources under the tenant. In this example, the tenant has a tenant ID of 2. So, a user with an IP of 10.20.30.5 can access the management API.

**User level example**

In the following example, the POST request creates a whitelist for a specific user. In cases where end users create their own data sources, administrators can allow end users to configure their own data access and Management API whitelists.

**Request**

POST https://MyServer:8443/api/mgmt/security/whitelist/user/1

**Request Payload**

{
 "managementAPI": [
 {   "startAddress": "10.20.30.2"
  }
 ],
"adminAPI": [
 {   "startAddress": "10.20.30.2"
  }
 ],
"dataAccess": [
 {   "startAddress": "10.20.30.2"
  }
 ]
}

If the above request is successful, the specified IP address is whitelisted for the specified user owned resources. So, a user with an IP address of 10.20.30.2 only can access the Management API.

**See also**

IP Address Whitelist API on page 1195
Enabling and disabling the IP address whitelist feature

You can use either the Web UI or the System Configurations API to enable or disable the IP address whitelist feature.

**Note:** The IP address whitelist feature is enabled by default. Unless you have disabled this feature, any IP address whitelists you create will immediately be enforced.

**Using the Web UI**

Take the following steps to enable or disable IP address whitelists.

1. Navigate to the **System Configurations** view by clicking the system configurations icon.
2. Toggle the **IP WhiteList Filtering** switch to the desired setting.
3. Click **Save** to save the change.

**Using the System Configurations API**

The following GET operation retrieves the current setting. The number 8 is the ID of the IP address whitelist feature.

```
GET https://<myserver>:<port>/api/admin/configurations/8
```

```
{
  "id": 8,
  "description": "Enable IP Whitelist filtering, when value is set to true. Default value is "true".
  "value": "true"
}
```

The following PUT request disables the IP address whitelist feature.

```
PUT https://<myserver>:<port>/api/admin/configurations/8
```

```
{
  "value": "false"
}
```

The following PUT request enables the IP address whitelist feature.

```
PUT https://<myserver>:<port>/api/admin/configurations/8
```

```
{
  "value": "true"
}
```

**See also**

[System Configurations API](#) on page 1124
Configuring row limit throttling

Hybrid Data Pipeline supports limiting query results to a specified number of rows (row limit throttling). Throttling is handled with the MaxFetchRows limit. The MaxFetchRows limit can be applied at four levels: system, tenant, user, and data source. The following hierarchy applies to these levels.

1. Data source
2. User
3. Tenant
4. System

A row limit set on a data source overrides any row limit set at the other levels; a row limit set on a user account overrides a row limit set on a tenant or at the system level; a row limit set on a tenant overrides a row limit set at the system level; and a row limit set at the system level overrides the default row limit. Default and system limits apply to behavior across Hybrid Data Pipeline, while limits on data sources, users, and tenants apply to the resources they handle.

To configure row limit throttling, the administrator must have either the Administrator (12) or the Limits (27) permission.

System level configuration

Row limit throttling can be configured at the system level either with the Web UI or with the Limits API. For details on using the Web UI, see Manage Limits view on page 76.

The following POST creates a system-level limit of 1000 rows. The number 1 is the ID of the MaxFetchRows limit. The payload passes 1000 as the value for this limit.

```
POST https://<myserver>:<port>/api/admin/limits/system/1

{   "value": 1000
}
```

Tenant configuration

Row limit throttling can be configured at the tenant level either with the Web UI or with the Limits API. For details on using the Web UI, see Manage Limits view on page 76.

The following POST sets a limit of 1500 rows on the specified tenant. The number 32 is the ID of the tenant, and the number 1 is the ID of the MaxFetchRows limit. The payload passes 1500 as the value for this limit.

```
POST https://<myserver>:<port>/api/admin/limits/tenants/32/1

{   "value": 1500
}
```

User account configuration

Row limit throttling can be configured at the user level either with the Web UI or with the Limits API. For details on using the Web UI, see Manage Users view on page 65.
The following POST sets a limit of 2000 rows on the specified user account. The number 86 is the ID of the user account, and the number 1 is the ID of the MaxFetchRows limit. The payload passes 2000 as the value for this limit.

```
POST https://<myserver>:<port>/api/admin/limits/users/86/1

[    
  "value": 2000
]
```

**Data source configuration**

Row limit throttling can only be configured at the data sources level using the Limits API. The following POST sets a limit of 2500 rows on the specified data source. The number 86 is the ID of the user account; the number 14 is the ID of the data source that is owned by the user account; and the number 1 is the ID of the MaxFetchRows limit. The payload passes 2500 as the value for this limit.

```
PUT https://<myserver>:<port>/api/admin/limits/users/86/datasources/14/1

[    
  "value": 2500
]
```

**See also**

- Manage Limits view on page 76
- Manage Users view on page 65
- Limits API on page 1071

## Configuring throttling for OData queries

Hybrid Data Pipeline supports throttling OData queries. By default, when executing an OData query, Hybrid Data Pipeline sends the query to the backend data store. All of the results are then fetched and persisted, and the first page of results is returned to the application. Having multiple large queries running simultaneously can consume system resources, which can negatively impact the performance of the service for all users. Further, some results are never fully viewed by applications, meaning that resources are unnecessarily allocated to return unused data. As a result, an administrator may want to limit the number of active OData queries to throttle the amount of resources consumed.

OData query throttling may be configured with the ODataMaxConcurrentQueries limit. When the ODataMaxConcurrentQueries limit is set to 0 (zero), there is no maximum number of persisted queries. When ODataMaxConcurrentQueries is set to a positive integer, rows are fetched one page in advance of application requests. This maintains quick response times in addition to minimizing the expense associated with executing large queries. Queries that contain more than one page of results are persisted in system memory until completely returned to the application or terminated. To prevent users from exhausting system and database resources, the maximum number of persisted queries are limited to the value specified. When this limit is exceeded, the least recently used query is canceled, and subsequent attempts to retrieve data from the canceled query will fail.

The ODataMaxConcurrentQueries limit can be applied at four levels: system, tenant, user, and data source. The following hierarchy applies to these levels.

1. Data source
2. User
3. Tenant
4. System

A query limit set on a data source overrides any query limit set at the other levels; a query limit set on a user account overrides a query limit set on a tenant or at the system level; a query limit set on a tenant overrides a query limit set at the system level; and a query limit set at the system level overrides the default query limit. Default and system limits apply to behavior across Hybrid Data Pipeline, while limits on data sources, users, and tenants apply to the resources they handle.

To configure OData query throttling, the administrator must have either the Administrator (12) or the Limits (27) permission.

**System level configuration**

OData query throttling can be configured at the system level either with the Web UI or with the Limits API. For details on using the Web UI, see Manage Limits view on page 76.

The following POST creates a system-level limit of 50000 queries. The number 6 is the ID of the ODataMaxConcurrentQueries limit. The payload passes 50000 as the value for this limit.

```
POST https://myserver:port/api/admin/limits/system/6

{   "value": 50000
}
```

**Tenant configuration**

OData query throttling can be configured at the tenant level either with the Web UI or with the Limits API. For details on using the Web UI, see Manage Limits view on page 76.

The following POST sets a limit of 60000 queries on the specified tenant. The number 32 is the ID of the tenant, and the number 6 is the ID of the ODataMaxConcurrentQueries limit. The payload passes 60000 as the value for this limit.

```
POST https://<myserver>:<port>/api/admin/limits/tenants/32/6

{   "value": 60000
}
```

**User account configuration**

OData query throttling can be configured at the user level either with the Web UI or with the Limits API. For details on using the Web UI, see Manage Users view on page 65.

The following POST sets a limit of 3000 queries on the specified user account. The number 86 is the ID of the user account, and the number 6 is the ID of the ODataMaxConcurrentQueries limit. The payload passes 3000 as the value for this limit.

```
POST https://<myserver>:<port>/api/admin/limits/users/86/1

{   "value": 3000
}
Data source configuration

OData query throttling can only be configured at the data sources level using the Limits API. The following POST sets a limit of 10000 queries on the specified data source. The number 86 is the ID of the user account; the number 14 is the ID of the data source that is owned by the user account; and the number 6 is the ID of the ODataMaxConcurrentQueries limit. The payload passes 10000 as the value for this limit.

```
POST https://<myserver>:<port>/api/admin/limits/users/86/datasources/14/1
{
    "value": 10000
}
```

See also

- Manage Limits view on page 76
- Manage Users view on page 65
- Limits API on page 107

Configuring CORS behavior

Hybrid Data Pipeline supports cross-origin resource sharing (CORS) filters that allow the sharing of web resources across domains. CORS provides several advantages over sites with a single-origin policy, including improved resource management and two-way integration between third-party sites. An administrator can enable or disable CORS filtering with the CORSBehavior limit. In turn, the CORS Whitelist API must be used to create and manage a whitelist of trusted origins.

CORS filtering can only be applied at the system level. It cannot be applied to individual tenants. To enable or disable CORS, the administrator must have either the Administrator (12), or the Limits (27) permission and administrative access on the default system tenant. To create and manage a whitelist, the administrator must have either the Administrator (12) permission, or the CORS whitelist (23) permission and administrative access on the default system tenant.

Enabling CORS behavior

CORS filters can be enabled either with the Web UI or with the Limits API. For details on using the Web UI, see Manage Limits view on page 76.

CORS filtering is disabled by default (CORSBehavior set to 0), and resources are shared only with pages of the same origin. CORS filtering can be enabled by setting the CORSBehavior limit to 1 or 2 via the Limits API. When CORSBehavior is set to 1, the CORS filter is enabled with all origins trusted. When CORSBehavior is set to 2, the CORS filter is enabled with a whitelist of trusted origins. The following POST operation specifies the CORSBehavior endpoint (5). The payload sets the CORSBehavior limit to 2.

```
POST https://myserver:port/api/admin/limits/system/5
{
    "value": 2
}
```

Creating a whitelist for CORS filtering

When CORS filtering has been enabled to use a whitelist of trusted origins (CORSBehavior set to 2), a whitelist must be created to complete a CORS configuration. The CORS Whitelist API must be used to create the whitelist of trusted origins. The following POST operation specifies the whitelist endpoint with a payload that specifies domains for the trusted origins.
Note: The wild card * can be used at the beginning of a domain. For example, *.progress.com is a valid entry, and will whitelist any origin that ends with progress.com. The wild card is not supported at any other location within a domain. For example, progress.abc.*.com is not supported for origin validation.

```json
POST https://<myserver>:<port>/api/admin/security/cors/whitelist

{
    "whitelist": [
        {
            "domain": "http://*.abc.com",
            "description": "The ABC group domain"
        },
        {
            "domain": "http://bar.test.com",
            "description": "The bar trusted origin"
        }
    ]
}
```

See also
- Manage Limits view on page 76
- Limits API on page 1071
- CORS Whitelist API on page 1063

### FIPS (Federal Information Processing Standard)

The Federal Information Processing Standard (or FIPS) is a cryptography standard created by the U.S. government. FIPS specifications require certain secure algorithms, cryptographic modules and random number generation. Hybrid Data Pipeline uses the Bouncy Castle libraries to provide FIPS 140-2 compliant cryptography. Using FIPS in Hybrid Data Pipeline server changes the following:

- The way we secure Pseudo-Random Number Generation for cryptographic elements
- The modules used for generating encrypted data including SSL
- The handling of SSL certificates, including the generation of the java truststore and keystore to be compatible with the Bouncy Castle libraries

Note: If you plan to run Hybrid Data Pipeline in FIPS mode and use a Java plugin to support external authentication services, the Java plugin must be FIPS compliant. In addition, the external authentication Java plugin should be tested with FIPS mode enabled before moving to a production environment.
Enabling and disabling FIPS

Configuring Hybrid Data Pipeline server for FIPS support

There are two ways to configure the Hybrid Data Pipeline server for FIPS support:

• Through an installer during the initial Hybrid Data Pipeline server installation. By default, Hybrid Data Pipeline will be installed in a FIPS disabled mode. You need to explicitly opt for FIPS support on the relevant installation screen.

• Using the script `enable_fips.sh`

Note: Before enabling FIPS, you must ensure that your hardware supports secure random, or you have a secure random daemon installed. For further details see Before enabling FIPS on page 176.

Note: We recommend a new, clean installation with FIPS enabled for production environments. With a new installation, users and datasources must be re-created. The script will not change the stored encryption keys which if generated by a non-FIPS install use the same encryption algorithm, but with the less secure random number generation.

Enable FIPS during installation

Before enabling FIPS, you must ensure that your hardware supports secure random, or you have a secure random daemon installed. To enable FIPS during installation, you must:

1. Run the installer, GUI or Console mode, and choose your desired options.
2. Choose Custom on the Install Type screen.
3. On the FIPS Configuration screen, check the Enable FIPS check-box.

Complete the remaining installation steps to install FIPS enabled Hybrid Data Pipeline server.
Enable FIPS after installation

Prerequisite: Before enabling FIPS, you must ensure that your hardware supports secure random, or you have a secure random daemon installed. To enable FIPS support after the installation:

1. Go to the installation directory, /Progress/DataDirect/Hybrid_Data_Pipeline/Hybrid_Server/ddcloud
2. Verify that the following two scripts exist for FIPS support:
   - disable_fips.sh
   - enable_fips.sh
3. Execute the enable_fips.sh script to enable FIPS support for the Hybrid Data Pipeline server. Note that running the script will force the Hybrid Data Pipeline Server to restart.

```
nc-hdp-u19:~/Progress/DataDirect/Hybrid_Data_Pipeline/Hybrid_Server/ddcloud% ./enable_fips.sh
```
4. After the script has completed, verify that FIPS is enabled. To verify, you can look at the standard output of the enable_fips.sh script. The final line output in a successful execution will be ‘Finished setting security provider’ and the script will exit with a return code of 0. If it fails, the appropriate error(s) will be displayed in the console, and the script will exit with a return code of 1.

Additionally, ./enable_fips.sh force can be run. By default enable_fips.sh will not attempt to generate the existing .bks Bouncy Castle keystore and truststore if FIPS compatibility is already enabled. With the optional force argument it forces both .bks Bouncy Castle keystore and truststore to be regenerated from the default Sun .jks files. If it is in a multimode install you will need to run enable_fips.sh on a single node, then restart the other nodes. The change will be detected on startup by the other Hybrid Data Pipeline nodes.

Disable FIPS

To disable FIPS:

1. Go to the installation directory, /Progress/DataDirect/Hybrid_Data_Pipeline/Hybrid_Server/ddcloud
2. Execute the disable_fips.sh script to enable FIPS support for the Hybrid Data Pipeline server.

```
nc-hdp-u19:~/Progress/DataDirect/Hybrid_Data_Pipeline/Hybrid_Server/ddcloud% ./disable_fips.sh
```
3. After the script has completed, verify that FIPS is disabled. To verify, you can look at the standard output of the enable_fips.sh script. The final line output in a successful execution will be ‘Finished setting security provider’ and the script will exit with a return code of 0. If it fails, the appropriate error(s) will be displayed in the console, and the script will exit with a return code of 1.

**Note:** Running the script will force the Hybrid Data Pipeline Server to restart.
Before enabling FIPS

FIPS support should only be enabled if the hardware on the server machine supports secure random. If FIPS support is enabled on a server machine that does not support secure random, the installer and the Hybrid Data Pipeline server may hang as they wait for the system to generate sufficiently random numbers for security-related tasks like encrypting or decrypting database information. To check if your hardware supports secure random on Intel hardware, you can examine the CPU flags to see if the `rdrand` instruction is supported.

```
$ cat /proc/cpuinfo | grep rdrand
flags : fpu vme de pse tsc msr pae mca cmov pat pse36
        cliflush dts mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp lm constant_tsc
        arch_perfmon pebs bts nopl xtopology tsc Reliable nonstop_tsc aperfmperf pni
        pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave
        avx f16c rdrand hypervisor lahf_lm ida arat epb pln pts dtherm fsgsbase
        smp
```

Hybrid Data Pipeline can be installed on hardware that does not support secure random but if this is done, there should be a secure random daemon installed to avoid the Hybrid Data Pipeline installer and server from being blocked waiting for secure random seed values. Another method of determining if the CPU supports secure random number generation is to obtain information about which CPU is being used with `cat /proc/cpuinfo` and then visiting the listed CPU manufacture's website to obtain information about the specific CPU.

If Your Hardware Does Not Support Secure Random

If your hardware does not support secure random, but you wish to test the FIPS compliant components of Hybrid Data Pipeline you can do so by modifying the configuration files provided. The resulting Hybrid Data Pipeline instance will generate the correct components but they will not be FIPS compliant. Make the modification as follows:

1. In the `install_dir/jre/lib/security/java.security.bcfips` file change the line
   
   `securerandom.source=file:/dev/random`

   to
   
   `securerandom.source=file:/dev/urandom`

2. Enable FIPS mode as normal. After installation scripts are provided for enabling and disabling the FIPS compliant security provider. These scripts will automatically restart the local Hybrid Data Pipeline server instance. In a clustered environment you will need to run this script on a single node, and then restart the other nodes which will pick up the changes at startup. The scripts are found in `install_dir/ddcloud` and are as follows:

   - `enable_fips.sh`: Enables Bouncy Castle as the FIPS compliant security provider
   - `disable_fips.sh`: Enables Sun as the security provider. This is not FIPS compliant

**Note:** To add certificates to the keystore and truststore for a FIPS compliant installation, you need to run the installer and perform an upgrade to specify a new PEM with all the needed certificates and chains.

Data source logging

Hybrid Data Pipeline provides data source logging to record user activity against data sources. Administrators can set logging levels for data sources through the Web UI or the Logging API. The resulting data source activity log can be used to troubleshoot issues.
Note: In addition to data source logging, a number of other system logs are generated. See System logs on page 197 for details.

See the following topics for information on using data source logging.

- Setting data source logging levels on page 177
- The data source activity log on page 181

### Setting data source logging levels

There are two basic data source logging levels: logging level and privacy level. The logging level determines the level of detail to be included in the data source activity log, while the privacy level determines the type of information that gets logged. These logging levels apply to all data sources. Non-relational data sources, such as Salesforce and Oracle Eloqua, include additional loggers that, when enabled, pass information related to the internal SQL engine to the data source activity log. Data source logging levels can be set either via the Web UI or the Logging API.

Note: Enabling and increasing logging levels may adversely impact performance. Therefore, best practices recommend that logging levels be restored to their defaults once an issue has been resolved.

See the following sections for more information about data source logging levels and how to set them.

- Logging level
- Privacy level
- Driver loggers
- Setting data source logging levels via the Web UI
- Setting data source logging levels with the Logging API

### Logging level

By default, logging level is set to CONFIG to track servlet and worker thread activity. This usually provides enough information to pinpoint an issue. The following table describes each of the valid settings for the logging level.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEVERE</td>
<td>Used to indicate system level problems that may require intervention.</td>
</tr>
<tr>
<td>WARNING</td>
<td>Possible severe situation, but problem probably averted.</td>
</tr>
<tr>
<td>INFO</td>
<td>Basic activity that probably always wants to be tracked.</td>
</tr>
<tr>
<td>CONFIG</td>
<td>Tracks servlet and worker thread activity.</td>
</tr>
<tr>
<td>FINE</td>
<td>Debug diagnostics.</td>
</tr>
</tbody>
</table>
### Privacy level

By default, privacy level is set to **AllowNone**. This is the most restrictive setting where neither user data nor SQL statements are logged. The following table describes each of the valid settings for the privacy level.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowNone</td>
<td>This is the most restrictive level. Here neither user data nor SQL statements are logged.</td>
</tr>
<tr>
<td>AllowSQL</td>
<td>This level allows the logging of SQL statements, but not input parameter values or result set column data.</td>
</tr>
<tr>
<td>AllowData</td>
<td>This is the least restrictive level. It allows SQL statements and any data values to be logged.</td>
</tr>
</tbody>
</table>

### Driver loggers

In addition to logging and privacy levels, driver loggers are available for non-relational data sources. These loggers are disabled by default. They can be enabled by setting a level of granularity from **SEVERE** to **FINEST**. When these loggers are enabled, information related to the internal SQL engine is passed to the data source activity log. This information can be useful in pinpointing and resolving issues. The following table describes the loggers available for non-relational data sources, such as Salesforce and Oracle Eloqua.

**Note:** Driver loggers are not available for standard relational data sources such as DB2, SQL Server, and Oracle.

<table>
<thead>
<tr>
<th>Logger</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL</td>
<td>Logs events associated with how the embedded SQL engine interacts with the data store and application.</td>
</tr>
<tr>
<td>Cloud</td>
<td>Logs JDBC spy calls to troubleshoot JDBC interactions between the connectivity service and the data store.</td>
</tr>
<tr>
<td>Driver Communication</td>
<td>Logs events associated with the JDBC calls made into the embedded SQL engine.</td>
</tr>
<tr>
<td>Adapter</td>
<td>Logs events related to how the connectivity service communicates with the data store in question.</td>
</tr>
</tbody>
</table>

### Setting data source logging levels via the Web UI

Either set of the following permissions are required to set logging levels through the Web UI.

- Administrator (12) permission
• WebUI (8) permission, Logging (24) permission, and administrative access on the tenant to which the users and data sources belong

Take the following steps to set logging levels via the Web UI.

1. Navigate to the Data Sources view by clicking the data sources icon.
2. For multitenant environments, select the tenant to which the user and data source belong.
3. Select the user who owns the data source.
4. Click the logging configurations icon next to the data source for which you are configuring logging. The Logging Settings page is displayed.
5. Set **Logging Level** and **Privacy Level** to desired level.
6. For non-relational data sources, enable driver loggers by setting the loggers to the desired level of granularity.
7. Click **Save**.

**Setting data source logging levels with the Logging API**

Either set of the following permissions are required to set logging levels through the Logging API.

- Administrator (12) permission
- Logging (24) permission and administrative access on the tenant to which the users and data sources belong

**Retrieve a user's data sources**

To retrieve the logging levels on a data source, the data source ID must be specified as a URL parameter. If you don't know the data source ID, you can execute the following GET operation to retrieve a list of data sources for a user. In this example, the number 9 is the user ID. The response payload follows the operation.

```plaintext
GET https://MyServer:8443/api/admin/users/9/datasources
```

```
{
    "dataSources": [
    {
        "id": 51,
        "name": "SF_test_ds_1",
        "dataStore": 1,
        "isGroup": false,
        "description": ""
    },
    {
        "id": 52,
        "name": "SF_test_ds_2",
        "dataStore": 1,
        "isGroup": false,
        "description": ""
    },
    {
        "id": 53,
        "name": "SF_test_ds_1",
        "dataStore": 1,
        "isGroup": false,
        "description": ""
    }
    ]
}
```

**Retrieve the logging levels of a data source**
You can now use the data source ID from the response payload to retrieve the logging configurations for the data source. The GET operation used to retrieve data source logging levels requires that you pass the user ID (9) and data source ID (51) as URL parameters, as in the following example. The response payload follows.

```
GET https://MyServer:8443/api/admin/users/9/datasources/51/logging

{
  "dasLogLevel": "CONFIG",
  "privacyLevel": "AllowNone",
  "driverLogConfig": [
    { "name": "ADAPTER", "logLevel": "OFF" },
    { "name": "CLOUD", "logLevel": "OFF" },
    { "name": "DRIVERCOMMUNICATION", "logLevel": "OFF" },
    { "name": "SQL", "logLevel": "OFF" }
  ]
}
```

**Update the logging levels of a data source**

An UPDATE operation can now be executed against the same endpoint to configure logging on the data source. As shown in the following example, a corresponding request payload provides the required configuration information.

```
PUT https://MyServer:8443/api/admin/users/9/datasources/51/logging

{
  "dasLogLevel": "CONFIG",
  "privacyLevel": "AllowSQL",
  "driverLogConfig": [
    { "name": "ADAPTER", "logLevel": "SEVERE" },
    { "name": "CLOUD", "logLevel": "SEVERE" },
    { "name": "DRIVERCOMMUNICATION", "logLevel": "SEVERE" },
    { "name": "SQL", "logLevel": "SEVERE" }
  ]
}
```
The data source activity log

The data source activity log records user activity against data sources. The data source activity log is written to the following directory where `install_dir` is the Hybrid Data Pipeline installation directory.

```
install_dir/ddcloud/das/server/logs/das
```

When running the server on multiple nodes behind a load balancer, a data source activity log is created for each instance of the service. In this scenario, multiple logs may need to be reviewed to identify errors, since the operation in question may have been handled by any one of the nodes.

The name of the data source activity log takes the following format.

```
[api][user_account][data_store][data_source_name].datestamp.log
```

For example:

```
[odbc][user123][oracle][oracle_odata_ds].2019-05-07.log
```

**Note:** For data sources using an On-Premises Connector, a corresponding data source activity log is generated on the machine hosting the connector. The name of the connector log file has the same format as the server log file. The connector data source activity log may be found in the following directory.

```
opc_install_dir\OPDAS\server\logs\das
```

The following sample shows that every entry in the data source activity log file starts out with the same standard information.

```
08-Sep-2017 07:11:54.493 CONFIG [http-bio-8080-exec-12] [steve@abctestmail.com] [salesforce] [d2c_salesforce_odatav4] [aYDHNNkB6F6mCk3]. [execute] [success=true] [ms=82 [stmtId=1] [bytesIn=2] [bytesOut=1861] [worker=Worker-1] [rowsFetched=0]
```

The following table can be used to parse the information contained in the data source activity log.

**Table 1: Data source activity log elements**

<table>
<thead>
<tr>
<th>Element</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamp</td>
<td>08-Sep-2017 07:11:54.493</td>
<td>UTC date time value for when the logging event was written.</td>
</tr>
<tr>
<td>Log Level</td>
<td>CONFIG</td>
<td>The Java logging level associated with the event.</td>
</tr>
<tr>
<td>Thread Name</td>
<td>http-bio-8080-exec-12</td>
<td>The name of the thread that logged the event.</td>
</tr>
<tr>
<td>User Name</td>
<td><a href="mailto:steve@abctestmail.com">steve@abctestmail.com</a></td>
<td>The name of the user.</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>salesforce</td>
<td>The name of the data source.</td>
</tr>
<tr>
<td>Session Token</td>
<td>aYDHNNkB6F6mCk3</td>
<td>The session identifier assigned to the user.</td>
</tr>
<tr>
<td>Element</td>
<td>Example</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Operation Context</td>
<td>execute</td>
<td>The operational context in which the event occurred. If a Tomcat servlet thread, this will identify the command. Other types of operations include: login, logout, upload, clear, and continue. A worker value indicates the operation is being done asynchronously in a worker thread. This is only done as part of an execute request.</td>
</tr>
<tr>
<td>Message</td>
<td>success=true ms=82</td>
<td>The rest of the line (or lines) will be the actual log message. Most messages are just key value pairs. Most messages include a success flag. When the flag is false, an error event message will usually proceed the message. The ms key gives the duration of the operation in milliseconds.</td>
</tr>
</tbody>
</table>

**Using third party JDBC drivers with Hybrid Data Pipeline**

Hybrid Data Pipeline supports the use of third party JDBC drivers. This feature gives customers the ability to integrate data stores for which Hybrid Data Pipeline does not currently have a built-in integration. The integration of a third party driver enables Hybrid Data Pipeline to expose backend data to JDBC, ODBC, and OData clients.

Three general steps should be followed to integrate a third party driver with Hybrid Data Pipeline environment.

1. The third party driver must be evaluated to verify compatibility with Hybrid Data Pipeline.
2. The third party driver must be integrated with the Hybrid Data Pipeline environment.
3. A data source must be created with the third party driver to access data on the backend data store.
Verifying a third party JDBC driver for Hybrid Data Pipeline compatibility

The Hybrid Data Pipeline verification tool should be used to verify whether a third party driver is compatible with Hybrid Data Pipeline. The following files and scripts will be used in the verification process. These files are located in the tools folder of either a Hybrid Data Pipeline server installation, or an On-Premises Connector installation.

- **jdbcVerificationTool.jar** - The JDBC driver verification tool.
- **config.properties** - This file must be updated with driver-specific information before running the verification tool. The following information must be updated in the config.properties file.

```java
# Configure the database URL
DBURL= database_url
# Configure the driver class name
CLASSNAME=classname
# Configure the user name
USER= username
# Configure the password
PASSWORD= password
# Configure the schema name
SCHEMA= schemaname
# Configure the comma separated table names
TABLES= tablename1, tablename2
# Configure the top term supported by the database. Supported top term keywords are {LIMIT, ROWNUM, FIRST, FETCHFIRST, TOP}
TOPTERM= topterm
# Configure the location of the third party driver files.
LOCATION= \default\location
```

- **jdbcVerificationTool.sh** - This shell script reads the config.properties file and runs the verification tool. This file is used to run the tool in Linux.
- **jdbcVerificationTool.bat** - This bat file reads the config.properties file and runs the verification tool. This file is used to run the tool in Windows.
- **datastore_profile_template.xml** - This is the template profile file for the third party JDBC Driver.

1. Navigate to the tools folder.
   a) For the Hybrid Data Pipeline service: <install_dir>/ddcloud/tools
   b) For an On-Premises Connector service: <opc_install_dir>	ools

2. Update the config.properties file with driver information.

3. Run the JDBC verification tool. Use the jdbcVerificationTool.sh file for Linux, or the jdbcVerificationTool.bat file for Windows.

4. Review the reports generated by the verification tool. The reports can be seen in the Reports folder.

The tool generates the following files:

- A summary report summarizes the findings of the verification tool. It conveys the percentage of test cases that succeeded, and provides an overview of warnings and exceptions. It will also indicate if any of the warnings and exceptions are critical. Here is an example of what a summary report looks like:

```
---
Summary of Results:---------------------------------------------------------------
Total:20
---
```
Succeeded:19  
Failed:1  
Pass Percentage:95%

---

Conclusion:
---

This driver is compatible and can be used in HDP. However some of the functionality will be affected due to the following failures.  
Found un-supported data types, respective columns will not be exposed via OData.  
Found columns with longer size than the supported column size in table 'GTABLE', the list of columns that will not be exposed via OData: LVCOL,LVARBINCOL.

- A verbose report provides information on a full range of test cases, including metadata and SQL queries. This report also details all the errors, warnings and exceptions. Here is an example of what a verbose report looks like:

  ---
  JDBC Metadata API Verification
  ---

  API: getMaxCatalogNameLength  
  [SUCCESS]Succeeded with Value:32

  API: getTypeInfo  
  [SUCCESS]Succeeded with Table:null  
  [SUCCESS][BIT][UNSIGNED_ATTRIBUTE]Received:false  
  [SUCCESS][BIT][TYPE_NAME]Received:BIT

  API: getColumns TABLE:GTABLE  
  [SUCCESS]Succeeded with Table:GTABLE  
  [SUCCESS][CHARCOL][COLUMN_DEF]Received:null  
  [SUCCESS][CHARCOL][COLUMN_NAME]Received:CHARCOL  
  [SUCCESS][LVCOL][DATA_TYPE]Received:-1  
  [FAILURE][LVCOL][COLUMN_SIZE]Failed with exception:Actual size is 2147483647 and supported size is 32768  
  [SUCCESS][BITCOL][COLUMN_DEF]Received:null  
  [SUCCESS][BITCOL][DATA_TYPE]Received:-7  
  [FAILURE][LVARBINCOL][COLUMN_SIZE]Failed with exception:Actual size is 2147483647 and supported size is 32768  
  [SUCCESS][DATECOL][COLUMN_DEF]Received:null  
  ...

  ---
  SQL Query Processing
  ---

  ODATA QUERY: SELECT  
  [SUCCESS]Succeeded with Query:SELECT T0.`CHARCOL`, T0.`VCHARCOL`, T0.`DECIMALCOL`, T0.`NUMERICCOL`, T0.`SMALLCOL` FROM `GTABLE` T0

  ODATA QUERY: COUNT  
  [SUCCESS]Succeeded with Query:SELECT count(*) FROM `GTABLE` T0
  ...

- A `<driverclass>.datastore-profile.xml` is generated. This file can be used to specify any changes that need to be made to the third party JDBC driver. If the user intends to create this file from scratch, it should be named in the given format: `<driverclass>.datastore-profile.xml`. In case of any changes, the updated file must be placed in the same location as the third party JDBC driver.
Integrating the third party JDBC driver into Hybrid Data Pipeline

After confirming that the third party JDBC driver is compatible, the driver can be integrated with the Hybrid Data Pipeline environment. The driver must be copied to the `drivers` folder. The location of the `drivers` folder varies depending on the Hybrid Data Pipeline environment.

- In a standalone installation, the driver must be copied to the following location:
  - `<install_dir>/ddcloud/keystore` if the default key location was selected during installation of the server
  - `<user_specified_location>/shared/drivers` if a non-default key location was specified during installation of the server

- In a load balancer installation, the driver must be copied to the `drivers` directory in the key location specified during the installation of the initial Hybrid Data Pipeline node.

- In an On-Premises Connector installation, the drivers must be updated in the On-Premises Connector Installation directory: `<opc_install_dir>\OPDAS\server\drivers`. The profile xml for the third party driver will still be read from the Hybrid Data Pipeline server.

After the third party driver has been integrated with the Hybrid Data Pipeline environment, you can create a data source to access backend data. If you attempt to create the JDBC data source without plugging in the driver, you will get an error. Data sources can be created either through the Web UI or through the Hybrid Data Pipeline API. For information on creating data sources through the Web UI, see Creating data sources with the Web UI on page 224 and JDBC parameters for third party drivers on page 292. For information on creating data sources through the Hybrid Data Pipeline API, see Data Sources API on page 1277.

Note: The current limitation is that there should not be any conflicts between the classes among various drivers. Multiple drivers cannot use different versions of the same library.

Configuring Hybrid Data Pipeline to authorize client applications using OAuth 2.0

Hybrid Data Pipeline supports OAuth 2.0 based authentication for OData APIs, in addition to basic authentication. OAuth 2.0 is not backwards compatible with OAuth 1.0 or 1.1.

Customers using client applications or third-party applications like Salesforce Connect and Power BI will be able to invoke Hybrid Data Pipeline OData access endpoints by passing in the required tokens as opposed to passing in username and password in basic authentication. This allows users to grant applications access to their OData endpoints without storing their user credentials in the application. Hybrid Data Pipeline supports OAuth based authorization for OData access endpoints for OData version 2 and version 4.

To integrate a client Application with Hybrid Data Pipeline using OAuth 2.0, an application developer can integrate an in-house OData application with OAuth 2.0. The following section lists the steps involved in achieving an OData connection with Hybrid Data Pipeline using OAuth.

Establishing an OData connection using OAuth 2.0

Take the following steps to establish OData connectivity using OAuth 2.0.
1. A client registers a client application with Hybrid Data Pipeline. See Register client application on page 186. Once the application is registered, the Hybrid Data Pipeline service will issue client credentials in the form of a client identifier and a client secret.

2. The application uses the Client ID and Client Secret to generate an access token. Depending on the type of grant flow, the sequence of steps here will be different. See OAuth grant flows on page 187. The application must also specify the scope of access. Hybrid Data Pipeline currently supports one high level scope: “Allow data access via OData.”

3. When the client application attempts to connect, Hybrid Data Pipeline prompts the end user for login credentials. If valid credentials are used, Hybrid Data Pipeline asks if the application should be allowed access to resource specified in scope.

4. If the application is authorized to access the resource specified in the scope, then Hybrid Data Pipeline sends the access token and refresh token to the client applications callback URL.

5. Client uses access token to access OData endpoint. Using the access token, the client application can make OData requests against Hybrid Data Pipeline resource.

6. If the access token expires, the application uses the Client ID, the Client Secret and the refresh token to generate a new access token.

**Note:** If you want third-party applications to use Hybrid Data Pipeline OData URL to pull data via OAuth 2.0, you will need to perform additional configuration steps to achieve the OAuth flow. Consult your third-party application documentation for information.

### Register client application

To support OAuth 2.0 authentication, you must register your application with Hybrid Data Pipeline. With the Client Application Registration API, you can register a client application with Hybrid Data Pipeline to generate a client ID and client secret. The client ID and client secret can then be used to generate tokens that enable applications to authenticate against Hybrid Data Pipeline with OAuth 2.0.

You must provide the following details while registering your application:

- Application Name
- Application description
- Redirect URLs: This is a user defined list of authorized URLs and can include one or more valid URLs. These URLs instruct Hybrid Data Pipeline where to provide the access token and refresh token to the application. These are the URLs that the application should redirect to, on successful authorization. You can enter multiple URLs, separated by commas.

When the request is executed, a client ID and a client secret are generated. The Client ID is a publicly exposed string that is used by the Hybrid Data Pipeline Service API to identify the application, and is also used to build authorization URLs that are presented to users. The Client Secret is used to authenticate the identity of the application to the service API when the application requests access to a user’s account, and must be kept private between the application and the API.

### OAuth 2.0 tokens

OAuth gives client applications restricted access to your data on a resource server. To allow access, an authorization server grants tokens to the client application in response to an authorization. Hybrid Data Pipeline generates three kinds of tokens.
Authorization Code: This code is generated as part of OAuth Authorization grant flow. The authorization server creates this token and passes it to the client application via the browser. This code is exchanged by the client application to obtain an access token and refresh token.

Access Token: Once the application has an access token, it may use the token to access the user's account via the API, limited to the scope of access, until the token expires or is revoked. The access token expires in 60 minutes. When an access token expires, using it to make a request from the API will result in an "Invalid Token Error". The duration of validity of an access token can be modified using the System Limit API. See Limits API on page 1071.

Refresh Token: If your access tokens expire, refresh tokens allow applications to continue to have access to users' accounts without the user continually re-authorizing the application. The refresh token must be stored securely within the application. You can use the refresh token to get a new access token from the server. The Refresh token will be used to generate an Access Token. Once issued by Hybrid Data Pipeline, the Refresh token remains valid until the user revokes it.

OAuth grant flows

While OAuth 2.0 defines several different grant types, Hybrid Data Pipeline currently supports the following grant flows.

- Authorization grant flow (UI-based): used with server-side applications
- Resource Owner Password credentials grant flow (non-UI based): used with trusted applications, such as those owned by the service itself.

Grant Type: Authorization Code

The authorization code grant type is the most commonly used because it is optimized for server-side applications, where source code is not publicly exposed, and Client Secret confidentiality can be maintained. This is a redirection-based flow, which means that the application must be capable of interacting with the user-agent (i.e. the user's web browser) and receiving API authorization codes that are routed through the user-agent. Now we will describe the authorization code flow:

Step 1: Authorization Code Link

First, the user is given an authorization code link that looks like the following:


Here is an explanation of the link components:

- https://cloud.hybriddatapipeline.com/oauth2/authorize: the API authorization endpoint
- client_id: The client id of the application in Hybrid Data Pipeline
- redirect_uri: Where the service redirects the user-agent after an authorization code is granted
- response_type: Specifies that your application is requesting an authorization code grant
- scope: Specifies the level of access that the application is requesting. Hybrid Data Pipeline currently supports one high level scope - "Allow data access via OData"

Step 2: User Authorizes Application

When the user clicks the link, they must first log in to the service, to authenticate their identity (unless they are already logged in). Then they will be prompted by the service to authorize or deny the application access to their account. If the user chooses to allow, the grant flow carries on with the next step. If the user chooses to deny, an error message will be displayed, specifying that "User denied OAuth access".
Step 3: Application Receives Authorization Code

If the user clicks "Authorize Application", the service redirects the user-agent to the application redirect URI, which was specified during the client registration, along with an authorization code. The redirect would look something like this:

https://hybriddatapipeline.com/callback?code=AUTHORIZATION_CODE

Step 4: Application Requests Access Token

The application requests an access token from the API, by passing the authorization code along with authentication details, including the client secret, to the API token endpoint. The parameters are sent in request body as form url encoded. The following is an example of a POST request to Hybrid Data Pipeline's token endpoint.

https://cloud.hybriddatapipeline.com/oauth2/token

POST Call
client_id=CLIENT_ID
client_secret=CLIENT_SECRET
grant_type=authorization_code
code=AUTHORIZATION_CODE
redirect_uri=CALLBACK_URL

Step 5: Application Receives Access Token

If the authorization is valid, the API will send a response containing the access token and a refresh token to the application. The entire response will look something like this:

{
  "access_token":"ACCESS_TOKEN",
  "token_type":"bearer",
  "expires_in":600,
  "refresh_token":"REFRESH_TOKEN"
}

The application is now authorized. It may use the token to access the user's account via the API, limited to the scope of access, until the token expires or is revoked. A refresh token can be used to request a new access token if the original access token has expired.

Grant Type: Resource Owner Password Credentials

With the resource owner password credentials grant type, the user provides their service credentials (username and password) directly to the application. The application uses the /oauth2/token endpoint to obtain an access token from the service.

The following details are required in this grant type:

• Client credentials
• grant_type
• scope (included in the request body)

This grant type should only be used with trusted applications since user credentials need to be shared with the client application.

Resource Owner Password Credentials Flow
After the user gives their credentials to the application, the application will then request an access token from the authorization server. To generate the proper OAuth2 token, you need to pass the payload as "application/x-www-form-urlencoded". The POST request must include the user credentials in the request body. The authorization should be set to No Auth before posting the payload.

```plaintext
grant_type:password
scope:api.access.odata
username:<username>
password:<password>
client_id: <clientid>
client_secret:<client secret>
```

After the user credentials provided are authenticated, the authorization server returns an access token to the application. Now the application is authorized.

**OAuth 2.0 endpoints**

You can use the Hybrid Data Pipeline endpoints to register a client application, view a list of registered applications, reset client credentials, revoke access to a registered application, and otherwise manage client application access to Hybrid Data Pipeline data sources using OAuth 2.0. OAuth endpoints are the URLs that you use to make OAuth authorization requests to Hybrid Data Pipeline. The following is the list of OAuth 2.0 endpoints:
## Integrating Hybrid Data Pipeline with a Google OAuth 2.0 authorization flow to access Google Analytics

Hybrid Data Pipeline must be integrated as a client application into a Google OAuth 2.0 authorization flow to create a data source for accessing Google Analytics. The following workflow outlines the tasks required to achieve this integration. The remaining topics in this section provide detailed instructions for these tasks.

### OAuth Endpoints

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>Endpoints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get list of OAuth registered applications</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications</td>
</tr>
<tr>
<td>Register OAuth application</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications</td>
</tr>
<tr>
<td>Get registered application by ID</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}</td>
</tr>
<tr>
<td>Update registered application by ID</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}</td>
</tr>
<tr>
<td>Delete registered application by ID</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}</td>
</tr>
<tr>
<td>Reset client secret of application</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}/reset</td>
</tr>
<tr>
<td>Get list of applications for which logged-in user has access</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/allowedapplications</td>
</tr>
<tr>
<td>Revoke access granted for the given application ID</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/allowedapplications/{id}</td>
</tr>
<tr>
<td>Generate access token and refresh token</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth2/token</td>
</tr>
<tr>
<td>Authorize token</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth2/authorize</td>
</tr>
</tbody>
</table>

For additional information, see OAuth API for configuring Hybrid Data Pipeline to authorize client applications on page 1373.
1. Hybrid Data Pipeline must be registered as a client application with the Analytics API in the Google Developer Console.

2. The OAuth applications API must be used to create an OAuth application object. The OAuth application object holds the client ID and secret provided by Google. This information allows Hybrid Data Pipeline to identify itself as a registered application with the Analytics API.

3. An OAuth profile must be created as part of the process of creating a data source on a Google Analytics data store. (Once a profile has been created, it may be selected by a user during the creation of subsequent data sources.)

   - If creating the data source through the Web UI, the user enters the name of the new OAuth profile in the **OAuth Profile Name** field. Then, the user clicks **Authorize with Google**. The user is redirected to Google where he or she must login to Google. When the user clicks **Accept**, Google sends access and refresh tokens to Hybrid Data Pipeline. The user is then returned to the Hybrid Data Pipeline interface to finish creating the data source.

   - If creating the data source with the Hybrid Data Pipeline API, the user must begin by obtaining OAuth access and refresh tokens from Google. Then, the user creates an OAuth profile object with the OAuth profile API. Once the OAuth profile has been created, the user can proceed with creating the data source using the Data Sources API.

**Registering Hybrid Data Pipeline as a client application with the Google Analytics API**

Hybrid Data Pipeline must be integrated as a client application into a Google OAuth 2.0 authorization flow to create a data source for accessing Google Analytics. Registering Hybrid Data Pipeline as a client application with the Analytics API is the first task in achieving this integration.

Take the following steps to register Hybrid Data Pipeline as a client application with the Analytics API.

1. Launch the Google Developer Console and log in with the appropriate Google account credentials.
2. Create a new project.
3. Click **Library** on the left. Then navigate to and click **Analytics API**. Then enable the Analytics API.
4. Click **Credentials** on the left. Then click **Credentials in APIs & Services**.
5. Under the **OAuth consent screen** tab, enter the required details and click **Save**.
6. Under the **Credentials** tab, click **Create credentials > OAuth client ID**.
7. Provide the following information in the **Create client ID** dialog.
   a) Select **Web application**.
   b) Enter an application name.
   c) Specify the Hybrid Data Pipeline URL (for example, https://hdp-test:8443) in the **Authorized JavaScript origins** field. The domain name must be fully qualified.
   d) Click **Create**.
8. Copy the client ID and secret key to a text editor. These credentials will be used to create an OAuth application object using the OAuth applications API.

What to do next:
Once Hybrid Data Pipeline has been successfully registered with the Analytics API, an administrator should proceed with the next task in integrating Hybrid Data Pipeline with a Google OAuth 2.0 authorization flow: Using the OAuth applications API to create an OAuth application object.

Using the OAuth applications API to create an OAuth application object

Once Hybrid Data Pipeline has been registered as a client application with the Google Analytics API, an administrator can proceed with creating an OAuth application object. The OAuth application object holds the client ID and secret provided by Google. This information allows Hybrid Data Pipeline to identify itself as a registered application with the Analytics API during the OAuth 2.0 authorization flow.

In a multitenant environment, an OAuth application object can be created for a particular tenant. When an OAuth application is created for the system tenant, it can be used by users in either the system tenant or a child tenant to create data sources on Google Analytics data stores. When an OAuth application is created for a child tenant, it can only be used by users in the child tenant to create data sources on Google Analytics data stores. Even though they will be able to view OAuth application objects that exist in child tenants, administrators who reside in the system tenant can only use the OAuth application object in the system tenant when creating their own data sources. An OAuth application object must be created for the system tenant to permit the creation of Google Analytics data sources by users, including administrators, in the system tenant.

The permissions required to create and modify OAuth application objects for Google Analytics data stores depend on the tenant in which the user resides and the tenants for which the user has administrative access. With the Administrator (12) permission, a user can create an OAuth application object in any tenant across the system. With the MgmtAPI (11) and OAuth (28) permissions, a user in the system tenant can create an OAuth application object for the system tenant. This user can also create OAuth application objects for tenants for which he or she has administrative access. With the MgmtAPI (11) and OAuth (28) permissions, a user in a child tenant can create an OAuth application object only in the tenant in which he or she resides.

POST operation

The POST operation to create an OAuth application object will have the following syntax.

```
POST https://<myserver>:<port>/api/mgmt/oauthapps
```

Payload definition

The payload used to create the OAuth application object can be defined as follows.

```json
{
    "name": "oauth_application_name",
    "dataStore": data_store_id,
    "tenantId": tenant_id,
    "description": "oauth_application_description",
    "clientId": "client_id",
    "clientSecret": "client_secret"
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth application object.</td>
<td>Required</td>
<td>The user-specified name of the OAuth application object. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store for which the OAuth application object is being created.</td>
<td>Required</td>
<td>The only data store which Hybrid Data Pipeline currently supports access to is Google Analytics. Therefore, the only valid value is the Google Analytics data store ID: 54.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the OAuth application and data store belong.</td>
<td>Optional</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth application object.</td>
<td>Optional</td>
<td>A description provided by the user.</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The OAuth client_id generated by Google when an application is registered with the Analytics API in the Google Developer Console.</td>
<td>Required</td>
<td>A valid client_id.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The OAuth client_secret generated by Google when an application is registered with the Analytics API in the Google Developer Console.</td>
<td>Required</td>
<td>A valid client_secret.</td>
</tr>
</tbody>
</table>

**Example**

The following POST operation creates the *TenantA OAuth app* object.

```plaintext
POST https://MyServer:8443/api/mgmt/oauthapps

Request payload
```
"clientSecret": "1912308409123890"
}

Response payload

Status code: 201
Successful response

{
  "id": "17",
  "name": "Tenant A OAuth app",
  "dataStore": 54,
  "tenantId": 303,
  "description": "Tenant A OAuth application object for Google Analytics",
  "clientId": "asdfjasdljfasdkgf",
  "clientSecret": "1912308409123890"
}

What to do next

Users may now proceed with creating an OAuth profile and a Google Analytics data source.

- If creating the OAuth profile and data source through the Web UI, proceed to the following topics.
  - Creating data sources with the Web UI on page 224
  - How to create a data source in the Web UI on page 225
  - Google Analytics parameters on page 298

- If creating the OAuth profile and data source through the Web UI, proceed to the following topics.
  - Using the OAuth profiles API to create an OAuth profile on page 194
  - Using the Data Sources API to create a Google Analytics data source on page 196

Using the OAuth profiles API to create an OAuth profile

If a user intends to use the Data Sources API to create data sources on a Google Analytics data store, the user must first create an OAuth profile with the OAuth profiles API. The OAuth profiles API permits Hybrid Data Pipeline access to Google Analytics through the creation of an OAuth profile object. The OAuth profile object holds OAuth access and refresh tokens that are initially supplied by Google. These tokens enable Hybrid Data Pipeline to access Google Analytics on behalf of the user. Before a user can create an OAuth profile, he or she must obtain these tokens from Google before executing the POST to create the OAuth profile.

OAuth profiles are created or selected for data sources, and a single OAuth profile can be used for multiple data sources on a Google Analytics data store. Since OAuth profiles are associated with data sources, a user must have the CreateDataSource (1) permission to create a profile.

Once the user has obtained the required access and refresh tokens from Google, he or she may proceed with a POST operation to create an OAuth profile.

POST operation

The POST operation to create an OAuth profile will have the following syntax.

POST https://<myserver>:<port>/api/mgmt/oauthprofiles
Payload definition

The payload used to create the OAuth profile can be defined as follows.

```
{
    "name": "oauth_profile_name",
    "oauthAppId": "oauth_application_id",
    "description": "oauth_profile_description",
    "accessToken": "access_token",
    "refreshToken": "refresh_token"
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth profile.</td>
<td>Required</td>
<td>The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;oauthAppId&quot;</td>
<td>The ID of the OAuth application object.</td>
<td>Required</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth profile.</td>
<td>Optional</td>
<td>A description provided by the user.</td>
</tr>
<tr>
<td>&quot;accessToken&quot;</td>
<td>The access token includes the credential information required to gain access to the Google Analytics API.</td>
<td>Optional</td>
<td>A valid access token.</td>
</tr>
<tr>
<td>&quot;refreshToken&quot;</td>
<td>The refresh token is used to generate new access tokens.</td>
<td>Required</td>
<td>A valid refresh token.</td>
</tr>
</tbody>
</table>

Example

The following POST operation creates the Google_User_1 profile.

POST https://MyServer:8443/api/mgmt/oauthprofiles

Request payload

```
{
    "name": "Google_User_1",
    "oauthAppId": 17,
    "description": "OAuth profile 1",
    "accessToken": "111c334445e55",
    "refreshToken": "222d88899966fa"
}
```

Response payload

```
{
    "id": 33,
    "name": "Google_User_1",
    "oauthAppId": 17,
    "description": "OAuth profile 1",
    "accessToken": "111c334445e55",
```
"refreshToken": "222d8889996fa"
}

**What to do next**

Once a user has created an OAuth profile with Google-supplied access and refresh tokens, the user can proceed with creating a Google Analytics data source with the Data Sources API.

**Using the Data Sources API to create a Google Analytics data source**

Once a user has created an OAuth profile, the user can use the Data Sources API to create a Google Analytics data source. The user must have the CreateDataSource (1) permission to create a data source. For example, the following POST operation creates the `GoogleAnalytics_Test` data source.

```plaintext
POST https://MyServer:8443/api/mgmt/datasources
```

**Example request payload**

```plaintext
{
  "name": "GoogleAnalytics_Test",
  "dataStore": 54,
  "description": "DS for testing GA profiles",
  "options": {
    "OAuthProfileId": "31",
    "ODataVersion": "4",
    "DefaultQueryOptions": "segmentId=-1;",
    "ConfigOptions": "defaultView=Progress - No Filters"
  }
}
```

**Example response payload**

```
Status code: 201
Successful response
```

```plaintext
{
  "id": 279,
  "name": "GoogleAnalytics_Test",
  "dataStore": 54,
  "description": "DS for testing GA profiles",
  "options": {
    "OAuthProfileId": "17",
    "ODataVersion": "4",
    "AuthenticationCode": "4/ABCDEFGHIJKLMO_PQRSTuvWXYZ-ABc2-abC3",
    "DefaultQueryOptions": "segmentId=-1;",
    "ConfigOptions": "defaultView=Progress - No Filters"
  }
}
```

**Troubleshooting**

This section includes several topics to help troubleshoot issues with Hybrid Data Pipeline. **Contact Technical Support** for additional assistance.
System logs

Hybrid Data Pipeline generates a number of log files to record events, activity, and other information. As described in Data source logging on page 176, the user activity log provides the information needed to resolve most user issues. However, some issues may warrant further investigation. In such a scenario, Progress technical support can help you retrieve and examine other system logs, as well as the user activity log.

Hybrid Data Pipeline system logging falls into three general categories.

- **Deployment logging**
- **Runtime logging**
- **On-Premises Connector logging**

**Note:** Deployment and runtime logs can be bundled into a compressed tar file by running the `install_dir/ddcloud/getlogs.sh` script. If running the server on multiple nodes, the `getlogs.sh` script must be run on each host machine. The name of the tar file will have the following format.

```
d2c_logs.datetimestamp.tar.gz
```

**Deployment logging**

The following log files can be useful when investigating problems that occur during installation or upgrade of the server.

- `<install_dir>/ddcloud/final.log`

  The `final.log` file provides the overall status of a Hybrid Data Pipeline server deployment. If no errors were received during the deployment process, the file will contain the message "Hybrid Data Pipeline deployment complete." If an error does occur during the deployment process, this file will contain a message that indicates where the deployment script encountered the error.

- `<install_dir>/ddcloud/error.log`

  The `error.log` file provides error and warning messages received during the deployment process. If any error is received during deployment, the error message, or exception, is logged to this file.

- `<install_dir>/ddcloud/deploy.log`

  The `deploy.log` file provides details on the deployment process. In particular, this log file contains all parameters used in the configuration of the Hybrid Data Pipeline server, as well as any modifications to the system database schema.

**Runtime logging**

Runtime logging includes Tomcat log files, Web UI log files, and service log files. Hybrid Data Pipeline server runtime logs can be found in the following directory, and its sub-directories.

- `<install_dir>/ddcloud/das/server/logs`

  **Tomcat log files**

  The following Apache Tomcat log files are written to the `<install_dir>/ddcloud/das/server/logs` directory. These log files may be useful in diagnosing issues that occur when trying to start the Hybrid Data Pipeline service.

  - `localhost.datestamp.log`
  - `catalina.datestamp.log`
  - `catalina.out`
• `manager.datestamp.log`
• `localhost_access_log.datestamp.txt`

**Web UI log files**

The following logs record issues that occur with the Web UI.

• `install_dir/ddcloud/das/server/logs/d2c-ui/d2c-ui.log`
• `install_dir/ddcloud/das/server/logs/d2c-service-api/d2c-service-api.log`

**Service log files**

A number of log files record activity that relates directly to the operation of the Hybrid Data Pipeline service. The following table lists all service logs. (The service logs include the data source activity log described in [Data source logging](#) on page 176.)

<table>
<thead>
<tr>
<th>File name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[background].datestamp.log</td>
<td>This log captures logging events from the background threads in Hybrid Data Pipeline.</td>
</tr>
<tr>
<td>clouddb.datestamp.log</td>
<td>This log captures exceptions from non-relational data sources.</td>
</tr>
<tr>
<td>das-monitor.datestamp.log</td>
<td>System statistics are logged every 60 seconds.</td>
</tr>
<tr>
<td>ddcloud.datestamp.log</td>
<td>The log for initialization and shutdown of the servlet.</td>
</tr>
<tr>
<td>extauth.datestamp.log</td>
<td>Logging related to any external authentication services configured for the Hybrid Data Pipeline instance.</td>
</tr>
<tr>
<td>[filter].datestamp.log</td>
<td>This log is used by our Tomcat filters. These include the authentication filter, IP address whitelist filter, and CORS filter.</td>
</tr>
<tr>
<td>[messaging].datestamp.log</td>
<td>Logging related to the Hybrid Data Pipeline service internal message queue.</td>
</tr>
<tr>
<td>user_data_source_info.datestamp.log</td>
<td>The log where a specific user's data source activity is captured. This is the data source activity log described in <a href="#">Data source logging</a> on page 176.</td>
</tr>
<tr>
<td>onpremise.datestamp.log</td>
<td>Activity related to making on-premises connections using the On-Premises Connector.</td>
</tr>
<tr>
<td>[system].datestamp.log</td>
<td>This log file captures any runtime logging events that cannot be associated with a user.</td>
</tr>
</tbody>
</table>

**On-Premises Connector logging**

When the On-Premises Connector is being used to connect to data sources, log files are written to the installation directory of the On-Premises Connector. For the most part these log files are analogous to the service log files generated for each instance of a Hybrid Data Pipeline server. These files are written to the `connector_install_dir\OPDAS\server\logs\das` directory.
In addition, an opacessor.datestamp file is written to directly to the connector_install_dir\OPDAS\server\logs directory. This log captures information on communications between the On-Premises Connector and the Hybrid Data Pipeline server. If a problem occurs where the On-Premises Connector is unable to communicate with the Hybrid Data Pipeline server, this log may help identify the issue.

MySQL Community Edition troubleshooting

Hybrid Data Pipeline uses MySQL Connector/J when connecting to MySQL Community Edition. During installation, if you opt for using MySQL Community Edition as an external database or as a data source, you are prompted to specify the location of the MySQL Connector/J driver. This allows the installer to integrate MySQL Connector/J into the Hybrid Data Pipeline environment. Subsequently, you may configure data sources that connect to a MySQL CE data store and execute queries with ODBC, JDBC, and OData applications.

Since MySQL Connector/J is a separate component, it may require configuration and maintenance apart from Hybrid Data Pipeline. Therefore, you should refer to MySQL Connector/J documentation for information on support, functionality, and maintenance. In addition, the Progress DataDirect Hybrid Data Pipeline Installation Guide provides a procedure for upgrading the MySQL Connector/J driver without reinstalling the Hybrid Data Pipeline server.

Out of memory errors

Hybrid Data Pipeline automatically generates an HPROF binary heap dump when an out of memory error occurs. The service generates the file java_date_time.hprof in the Hybrid Data Pipeline installation directory whenever an out of memory error occurs. For example, install_dir/ddcloud/heapdumps/java_20170906_134157.hprof. The HPROF heap dump may contain sensitive information and should be handled securely.

Progress technical support will use the HPROF heap dump to help you analyze and resolve the out of memory error. When an out of memory error occurs, the Hybrid Data Pipeline service should be restarted.

IP address troubleshooting

Hosted database systems may by default limit client access based on IP addresses. Therefore, to access data using the Hybrid Data Pipeline service, you may need to modify security settings in your hosted environment to include the public IP address Hybrid Data Pipeline uses to access the database.

For example, if you wanted Hybrid Data Pipeline to access a database hosted on Amazon RDS, you would need to modify the default settings of your VPC security group to include the Hybrid Data Pipeline public IP address. In a Salesforce environment, you might similarly modify Trusted IP ranges for an organization.

Refer to vendor documentation regarding client access based on IP addresses for information on how to modify security settings.

Extracting schema files for non-relational data sources

In addition to providing connectivity to relational databases, Hybrid Data Pipeline offers connectivity to non-relational data stores, such as Salesforce and Oracle Service Cloud web services, that expose an object model. When creating a data source on a non-relational data store, Hybrid Data Pipeline generates map files that expose objects and fields as tables and columns. These map files can be used to develop SQL statements, better understand native metadata, and resolve issues in a given application environment. The Driver Files API can be used to obtain these map files.
Users can execute a GET operation on the following endpoints to obtain the files involved in the relational mapping of the object model.

- https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles
- https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles/native
- https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles/config

**Note:** To use these endpoints, the user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ViewDataSource (2) permission on the applicable data source.

When executing a GET operation on the /export/driverfiles endpoint, the response file is streamed to the users, who can download all the artifacts as a zip file. When executing a GET operation on the /export/driverfiles/config and /export/driverfiles/native endpoints, the entire file is returned as an XML response.

**See also**
- Export driver files for data source on page 1362
- Export config files for data source on page 1363
- Export native file for data source on page 1364

**Contacting Technical Support**

Progress DataDirect offers a variety of options to meet your support needs. Please visit our Web site for more details and for contact information:

https://www.progress.com/support

The Progress DataDirect Web site provides the latest support information through our global service network. The SupportLink program provides access to support contact details, tools, patches, and valuable information, including a list of FAQs for each product. In addition, you can search our Knowledgebase for technical bulletins and other information.

When you contact us for assistance, please provide the following information:

- Your number or the serial number that corresponds to the product for which you are seeking support, or a case number if you have been provided one for your issue. If you do not have a SupportLink contract, the SupportLink representative assisting you will connect you with our Sales team.
- Your name, phone number, email address, and organization. For a first-time call, you may be asked for full information, including location.
- The Progress DataDirect product and the version that you are using.
- The type and version of the operating system where you have installed your product.
- Any database, database version, third-party software, or other environment information required to understand the problem.
- A brief description of the problem, including, but not limited to, any error messages you have received, what steps you followed prior to the initial occurrence of the problem, any trace logs capturing the issue, and so on. Depending on the complexity of the problem, you may be asked to submit an example or reproducible application so that the issue can be re-created.
- A description of what you have attempted to resolve the issue. If you have researched your issue on Web search engines, our Knowledgebase, or have tested additional configurations, applications, or other vendor products, you will want to carefully note everything you have already attempted.
• A simple assessment of how the severity of the issue is impacting your organization.
Using Hybrid Data Pipeline

For details, see the following topics:

- Logging in to the Web UI
- Using Hybrid Data Pipeline APIs
- Using the Web UI
- Creating data sources with the Web UI
- Editing, testing, and deleting data sources with the Web UI
- Configuring data sources for OData connectivity and working with data source groups
- Creating and using REST data sources

Logging in to the Web UI

Logging in to the Web UI is a two step process. First, you must enter the URL of your Hybrid Data Pipeline instance in the address field of a supported browser. Then, you must enter your username and password at the Hybrid Data Pipeline login screen.

A URL includes the Web protocol, a server name, and a port number. For example:

https://MyServer:8443/hdpui

The syntax for this URL can be described as follows.

webprotocol://servername:portnumber
where

webprotocol

is the Web protocol, such as HTTP or HTTPS, used to connect to your Hybrid Data Pipeline instance.

servername

is the name of the machine hosting the Hybrid Data Pipeline service, or the name of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service.

portnumber

is the port number of the machine hosting the Hybrid Data Pipeline service, or the port number of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service. For a standalone installation, the port number is specified as the Server Access Port during installation. For a load balancer installation, the port number must be either 80 for http or 443 for https. Whenever port 80 or 433 are used, it is not necessary to include the port number in the URL.

See also
Initial login with default user accounts on page 58
Using the Web UI on page 63
Using Hybrid Data Pipeline APIs on page 62

Using Hybrid Data Pipeline APIs

Hybrid Data Pipeline provides a representational state transfer (REST) application programming interface (API) for managing Hybrid Data Pipeline connectivity service resources.

Hybrid Data Pipeline APIs use HTTP Basic Authentication to authenticate user accounts. The Hybrid Data Pipeline user ID and password are encoded in the Authorization header. The Hybrid Data Pipeline user specified in the Authorization header is the authenticated user.

To execute REST calls, you must pass a valid REST URL and pass a valid username and password to authenticate with basic authentication. A REST URL must include a base and resource-specific information. The base includes the Web protocol, a server name, and a port number, while resource-specific information provides a path to a particular resource necessary for performing an API operation. For example:

https://MyServer:8443/api/mgmt/datasources

**Note:** The port number is only required if the Hybrid Data Pipeline server or load balancer is configured to use a port other than 443 for SSL or 80 for non-SSL connections.

The syntax for a REST URL can be described as follows.

webprotocol://servername:portnumber/resourceinfo

where

webprotocol

is the Web protocol, such as HTTP or HTTPS, used to connect to your Hybrid Data Pipeline instance.
servername

is the name of the machine hosting the Hybrid Data Pipeline service, or the name of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service.

portnumber

is the port number of the machine hosting the Hybrid Data Pipeline service, or the port number of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service. For a standalone installation, the port number is specified as the Server Access Port during installation. For a load balancer installation, the port number must be either 80 for http or 443 for https. Whenever port 80 or 433 are used, it is not necessary to include the port number in the URL.

resourceinfo

is resource-specific information that provides a path to a particular Hybrid Data Pipeline resource necessary to perform an API operation.

See also

Hybrid Data Pipeline API reference on page 1037
Initial login with default user accounts on page 58
User provisioning on page 107
Logging in to the Web UI on page 61

Using the Web UI

The Hybrid Data Pipeline Web UI consists of views which can be selected from the navigation bar to the left. Access to these views, and the ability to execute operations they support, depend on permissions granted to the user (see Permissions and default roles on page 59 for details). These views include:

• Manage Tenants
• Manage Users
• Manage Roles
• Data Sources
• SQL Editor
• Manage Limits
• Manage External Authentication
• System Configurations

See the following topics for details on these views and other features of the Web UI.

• Manage Tenants view on page 64
• Manage Users view on page 65
• Manage Roles view on page 67
• Data Sources view on page 69
Manage Tenants view

The Manage Tenants view provides a list of tenants with description and status information for each tenant. With the appropriate permissions, you can add, modify, and delete tenants using this view.

The Manage Tenants view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) and TenantAPI (25) permissions, and administrative access on tenants the user administers

The following table provides permissions and descriptions for each action in the Manage Tenants view.

Note: Any user with Administrator (12) permission may perform all actions.
<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
</table>
| Create new tenant    | Web UI (8) TenantAPI (25)                        | To create a new tenant, click + New Tenant. Define the tenant with settings under each of the following tabs.  
  • General tab. Enter values in the given fields. The tenant name is required.  
  • Roles tab. Import roles from the parent tenant, if desired.  
  • Limits tab. Set throttling limits, if desired. |
| Edit a tenant        | Administrative access for the tenant Web UI (8) TenantAPI (25) | To edit a tenant, select a tenant from the list of tenants. Then, select Edit from the Actions dropdown. Edit the tenant settings as desired. |
| Delete a tenant      | Administrative access for the tenant Web UI (8) TenantAPI (25) | To delete a tenant, select the tenant you want to delete. Then, select Delete from the Actions dropdown. Confirm or cancel the delete operation in the dialog. |
| View tenant users    | Administrative access for the tenant Web UI (8) ViewUsers (14) TenantAPI (25) | To view the users of a tenant, select the tenant from the list of tenants. Then, select View Users from the Actions dropdown. You are directed to the Manage Users view where a list of users belonging to the tenant is displayed. See Manage Users view on page 65 for details. |
| Transfer tenant users| Administrative access for the system tenant and the tenant to which user(s) will be transferred Web UI (8) ViewUsers (14) ModifyUsers (15) TenantAPI (25) | To transfer users from the system tenant to a child tenant, select the child tenant from the list of tenants. Then, select Transfer Users from the Actions dropdown. You are directed to the Transfer User From System Tenant page. Select each user you want to transfer to the child tenant, and choose a role for each user from the role dropdown.  
  **Note:** Users can only be transferred from the system tenant to a child tenant. |

**Manage Users view**

The Manage Users view provides a list of users with roles and status information for a given tenant. With the appropriate permissions, you can add, update, and delete users using this view.

The Manage Users view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, ViewUsers (14) permission, ViewRole (18) permission, and administrative access on the tenant to which the users belong
The following table provides permissions and descriptions for each action in the Manage Users view.

**Note:** Any user with Administrator (12) permission may perform all actions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
</table>
| Filter users by tenant  | Administrative access to multiple tenants  
Web UI (8)  
ViewUsers (14)  
ViewRole (18) | An administrator with administrative access to multiple tenants will have the option of selecting the tenant for which he or she wants to view or manage users. Select the tenant for which you want to view users from the Select Tenant dropdown. |
| Create a new user       | Administrative access for the tenant  
Web UI (8)  
CreateUsers (13)  
ViewUsers (14)  
ViewRole (18) | To create a new user, click + New User. Define the user with settings under each of the following tabs.  
- **General** tab. Enter values in the given fields. User name and role are required.  
- **Authentication Setup** tab. The required information depends on the type of authentication you are using. See Authentication on page 143 for details.  
- **Limits** tab. Set throttling limits, if desired.  
- **Tenant Admin Access** tab. Grant the user administrative access to tenant(s), if desired. |
<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Edit a user**                  | Administrative access for the tenant  
Web UI (8)  
ViewUsers (14)  
ModifyUsers (15)  
ViewRole (18) | To edit a user, select a user from the list of users. Then, select **Edit** from the **Actions** dropdown. Edit user settings as desired.                                                                     |
| **Delete a user**                | Administrative access for the tenant  
Web UI (8)  
ViewUsers (14)  
DeleteUsers (16)  
ViewRole (18) | To delete a user, select the user you want to delete. Then, select **Delete** from the **Actions** dropdown. Confirm or cancel the delete operation in the dialog.                                           |
| **View the data sources owned by a user** | Administrative access for the tenant  
Web UI (8)  
ViewUsers (14)  
ViewRole (18)  
OnBehalfOf (21) | To view the data sources owned by a user, select a user from the list of users. Then, select **Data Sources** from the **Actions** dropdown. A list of data sources owned by the user is displayed. |

**Manage Roles view**

The **Manage Roles** view provides a list of roles for a given tenant. With the appropriate permissions, you can add, update, and delete roles using this view.

The **Manage Roles** view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, ViewRole (18) permission, and administrative access on the tenant to which the role(s) belong
The following table provides permissions and descriptions for each action in the Manage Roles view.

**Note:** Any user with Administrator (12) permission may perform all actions.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
</table>
| Filter roles by tenant  | Administrative access to multiple tenants  
Web UI (8)  
ViewRole (18) | An administrator with administrative access to multiple tenants will have the option of selecting the tenant for which he or she wants to view or manage roles. Select the tenant for which you want to view roles from the Select Tenant dropdown. |
| Create a new role       | Administrative access for the tenant  
Web UI (8)  
CreateRole (17)  
ViewRole (18) | To create a new role, click + New Role. Provide a name and description for the new role. Then, select permissions to define the role. |
<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit a role</td>
<td>Administrative access for the tenant</td>
<td>To edit a role, select the role from the list of roles. Then, select <strong>Edit</strong> from the <strong>Actions</strong> dropdown. Edit the role as desired.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewRole (18)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ModifyRole (19)</td>
<td></td>
</tr>
<tr>
<td>Delete a role</td>
<td>Administrative access for the tenant</td>
<td>To delete a role, select the role you want to delete. Then, select <strong>Delete</strong> from the <strong>Actions</strong> dropdown. Confirm or cancel the delete operation in the dialog.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewRole (18)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DeleteRole (20)</td>
<td></td>
</tr>
</tbody>
</table>

### Data Sources view

The **Data Sources** view allows you to manage data sources and data source groups. The **Data Sources** view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) and ViewDataSource (2) permissions

The **Data Sources** view consists of the following pages.

- **Data Sources**
- **Data Source Groups**

### Data Sources

The **Data Sources** page enables you to create, edit, and delete data source definitions. A data source definition configures the connection between Hybrid Data Pipeline and a data store.
The following table provides permissions and descriptions for basic actions in the Data Sources page. For detailed information on creating data sources, see Creating data sources with the Web UI on page 224 and How to create a data source in the Web UI on page 225.

**Note:** With the appropriate permissions, administrators can view data sources owned by other users through the Web UI. However, administrators cannot create, modify, or delete data sources owned by other users through the Web UI. To create, modify, or delete data sources that belong to other users, administrators must use Hybrid Data Pipeline APIs. See Data Sources API on page 1277 and Managing resources on behalf of users on page 1281 for further details.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter data sources by tenant</td>
<td>Administrative access to multiple tenants</td>
<td>An administrator with administrative access to multiple tenants will have the option of filtering by tenants to view data sources owned by a given user. Select the tenant in which the user resides from the Select Tenant dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong> Any user with the</td>
<td>Administrator (12)</td>
<td></td>
</tr>
<tr>
<td>Administrator (12) permission</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>can view the data sources of any user across all tenants</td>
<td></td>
</tr>
<tr>
<td>Action</td>
<td>Permissions</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Filter data sources by user</td>
<td>Administrative access to the tenant</td>
<td>To filter data sources by user, select the user whose data sources you want to view from the <strong>Select User</strong> dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Any user with the Administrator (12) permission can view the data sources of any user across all tenants.</td>
<td></td>
</tr>
<tr>
<td>Search for a data source</td>
<td>Web UI (8)</td>
<td>Use the search field in the upper right to filter data sources by name, data store, and description.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td>Create a new data source</td>
<td>Web UI (8)</td>
<td>To create a new data source, click <strong>+ New Data Source</strong>. See <strong>How to create a data source in the Web UI</strong> on page 225 for details.</td>
</tr>
<tr>
<td></td>
<td>CreateDataSource (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td>Modify a data source</td>
<td>Web UI (8)</td>
<td>To modify a data source, select the data source from the list of data sources. Then, select <strong>Edit</strong> from the <strong>Actions</strong> dropdown. Edit the data source as desired.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ModifyDataSource (3)</td>
<td></td>
</tr>
<tr>
<td>Delete a data source</td>
<td>Web UI (8)</td>
<td>To delete a data source, select the data source you want to delete. Then, select <strong>Delete</strong> from the <strong>Actions</strong> dropdown. Confirm or cancel the delete operation in the dialog.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DeleteDataSource (4)</td>
<td></td>
</tr>
<tr>
<td>Test a data source</td>
<td>Web UI (8)</td>
<td>To run queries against a data source through the Web UI, select the data source. Then, select <strong>SQL Testing</strong> from the <strong>Actions</strong> dropdown. You are directed to the <strong>SQL Editor</strong> view where you review schema and execute a SQL statement against the data source.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SQLEditorWebUI (10)</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>At least one of the following query permissions:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• UseDataSourceWithJDBC (5)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• UseDataSourceWithODBC (6)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• UseDataSourceWithOData (7)</td>
<td></td>
</tr>
</tbody>
</table>
Sync OData Schema

Web UI (8)
ViewDataSource (2)
ModifyDataSource (3)
MgmtAPI (11)

OData enabled data sources maintain an OData model. The OData model should be refreshed whenever the schema of the data source schema has been changed. To refresh the OData model, click the sync icon. For details, see Configuring data sources for OData connectivity and working with data source groups on page 622.

Obtain OData URI

Web UI (8)
ViewDataSource (2)

To obtain the OData URI for an OData enabled data source, copy the link associated with the link icon.

Configure data source logging

Web UI (8)
ViewDataSource (2)
Logging (24)

To configure data source logging, click the settings icon. You are directed to the Logging Settings page. Set logging and privacy levels as desired.

Data Source Groups

The Data Source Groups page enables you to combine OData enabled data sources into a single data source group. You can create, edit, and delete data source groups from this page.

The following table provides permissions and descriptions for basic actions in the Data Source Groups page. For detailed information on creating OData enabled data sources and data source groups, see Configuring data sources for OData connectivity and working with data source groups on page 622.
<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter data source groups by tenant</td>
<td>Administrative access to multiple tenants</td>
<td>An administrator with administrative access to multiple tenants will have the option of filtering by tenants to view data source groups owned by a given user. Select the tenant in which the user resides from the <strong>Select Tenant</strong> dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td><strong>Note:</strong> Any user with the Administrator (12) permission can view the data source groups of any user across all tenants.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td></td>
</tr>
<tr>
<td>Filter data source groups by user</td>
<td>Administrative access to the tenant</td>
<td>To filter data source groups by user, select the user whose data source groups you want to view from the <strong>Select User</strong> dropdown.</td>
</tr>
<tr>
<td></td>
<td>Web UI (8)</td>
<td><strong>Note:</strong> Any user with the Administrator (12) permission can view the data source groups of any user across all tenants.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewUsers (14)</td>
<td></td>
</tr>
<tr>
<td>Search for a data source group</td>
<td>Web UI (8)</td>
<td>Use the search field in the upper right to filter data source groups by name, data store, and description.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td>Create a new data source group</td>
<td>Web UI (8)</td>
<td>To create a new data source group, click <strong>+ New Group</strong>. See <strong>Creating a data source group</strong> on page 635 for details.</td>
</tr>
<tr>
<td></td>
<td>CreateDataSource (1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td>Modify a data source group</td>
<td>Web UI (8)</td>
<td>To modify a data source group, select the group. Then, select <strong>Edit</strong> from the <strong>Actions</strong> dropdown. Edit the group as desired.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ModifyDataSource (3)</td>
<td></td>
</tr>
<tr>
<td>Action</td>
<td>Permissions</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Delete a data source group</td>
<td>Web UI (8)</td>
<td>To delete a data source group, select the group you want to delete. Then, select <strong>Delete</strong> from the <strong>Actions</strong> dropdown. Confirm or cancel the delete operation in the dialog.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DeleteDataSource (4)</td>
<td></td>
</tr>
<tr>
<td>Obtain OData URI</td>
<td>Web UI (8)</td>
<td>To obtain the OData URI of a data source group, copy the link associated with the link icon 🌐.</td>
</tr>
<tr>
<td></td>
<td>ViewDataSource (2)</td>
<td></td>
</tr>
</tbody>
</table>

### SQL Editor view

The **SQL Editor** view allows users to browse schemas\(^4\) and to query data associated with a data source.

The **SQL Editor** view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, ViewDataSource (2) permission, SQLEditorWebUI (10) permission, and, to query data sources, at least one of the following query permissions:
  - UseDataSourceWithJDBC (5)
  - UseDataSourceWithODBC (6)
  - UseDataSourceWithOData (7)

\(^4\) For backend data stores that support schemas, the **Metadata Exposed Schemas** option can be used to restrict the exposed schemas to a single schema. **Metadata Exposed Schemas** only affects the metadata that is displayed in the **Schema** navigation pane. SQL queries can still be executed against tables in other schemas. For details, see the parameters topic for your data source type.
The following table provides permissions and descriptions for actions in the SQL Editor view. To perform any action from this view, begin by selecting a data source from the Select a Data Source dropdown.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Explore the schema and tables associated with the data source** | WebUI (8)  
ViewDataSource (2)  
SQLEditorWebUI (10) | To begin, a data source must be selected from the Select a Data Source dropdown. To view schema tables, click the a schema carrot in the Schema Tree panel. Click on a table to view the details of a table in the Table Details panel. Views and procedures that reside in the schema may also be listed. |
| **Execute a SQL statement against the data source** | Web UI (8)  
ViewDataSource (2)  
SQLEditorWebUI (10)  
At least one of the following query permissions:  
- UseDataSourceWithJDBC (5)  
- UseDataSourceWithODBC (6)  
- UseDataSourceWithOData (7) | To begin, a data source must be selected from the Select a Data Source dropdown. To run a query against the data source, enter the SQL statement in the field provided in the Editor panel. Then click Execute to run the query. SQL query results will be returned in the Results panel.  

**Note:** Queries made via the SQL Editor view time out after 6 minutes. Therefore, to validate a data source connection, you should execute queries that require less processing time. For large queries, only the first 200 results are shown.
Manage Limits view

The Manage Limits view allows you to view and set limits for features such as throttling and logging.

In the Manage Limits view, limits can be set at either the system or tenant level. System limits apply to behavior across Hybrid Data Pipeline and override default behavior, while tenant limits apply to the resources of a given tenant and override default behavior and system limits. Most limits can only be configured at the system level. However, some limits, such as MaxFetchRows and MaxConcurrentQueries, can be configured at any level.

**Note:** Limits can also be specified for users and data sources. User limits can be set either through the Manage Users view on page 65 or the Limits API on page 1071. User limits override default, system, and tenant limits. Data source limits can only be set via the Limits API on page 1071. Data source limits override all other limits.

The Manage Limits view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, Limits (27) permission, and administrative access on the given tenant

The table below provides descriptions for limits that may be set via the Manage Limits view.

**Note:**

- Throttling limits can be set either for the system tenant or any child tenant across the system.
- Log Management, Data Usage Meter, and Security limits can only be set for the system.
- To set system limits, the system tenant must be selected from the Tenant dropdown. The user must have the Administrator (12) permission.
• To set tenant limits, the child tenant must be selected from the **Tenant** dropdown. The user must have either the Administrator (12) permission, or WebUI (8), Limits (27) permissions, and administrative access on the given tenant.

• Tenant limits can also be set via the **Manage Tenants view** on page 64.

<table>
<thead>
<tr>
<th>Category</th>
<th>Limit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throttling</td>
<td>MaxFetchRows</td>
<td>Maximum number of rows allowed to be fetched for a single query.</td>
</tr>
<tr>
<td>Throttling</td>
<td>ODataMaxConcurrentQueries</td>
<td>Maximum number of concurrent active OData queries per data source.</td>
</tr>
<tr>
<td>Throttling</td>
<td>TransactionTimeout</td>
<td>The number of seconds the system allows a transaction to be idle before rolling it back.</td>
</tr>
<tr>
<td>Throttling</td>
<td>XdbcMaxResponse</td>
<td>Approximate maximum size of JDBC/ODBC HTTP result data in KB.</td>
</tr>
<tr>
<td>Log Management</td>
<td>LogRetentionDays</td>
<td>Number of days log files should be retained.</td>
</tr>
<tr>
<td>Log Management</td>
<td>MonitorRetentionDays</td>
<td>Number of days monitor details should be retained</td>
</tr>
<tr>
<td>Data Usage Meter</td>
<td>UserMeterRetentionDays</td>
<td>Number of days user meter details should be retained</td>
</tr>
<tr>
<td>Data Usage Meter</td>
<td>UserMeterWriteInterval</td>
<td>The number of seconds the system waits before scanning sessions for current metrics. A lower setting will result in more rows written to the meter table.</td>
</tr>
<tr>
<td>Data Usage Meter</td>
<td>UserMeterMaxAge</td>
<td>The number of seconds the system waits before writing out meter records. A lower setting will result in the rows written to the meter table to occur more frequently.</td>
</tr>
<tr>
<td>Security</td>
<td>PasswordLockoutInterval</td>
<td>The duration, in seconds, for counting the number of consecutive failed authentication attempts.</td>
</tr>
<tr>
<td>Security</td>
<td>PasswordLockoutLimit</td>
<td>The number of consecutive failed authentication attempts that are allowed before locking the user account.</td>
</tr>
<tr>
<td>Security</td>
<td>PasswordLockoutPeriod</td>
<td>The duration, in seconds, for which a user account will not be allowed to authenticate to the system when the PasswordLockoutLimit is reached.</td>
</tr>
<tr>
<td>Security</td>
<td>OAuthAccessTokenDuration</td>
<td>The duration, in minutes, for which a Access token is valid.</td>
</tr>
<tr>
<td>Category</td>
<td>Limit</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Security</td>
<td>OAuthAccessTokenCacheSize</td>
<td>Number of OAuth access tokens to be cached in memory for OAuth Authentication. By default up to 2000 tokens will be cached in memory.</td>
</tr>
<tr>
<td>Security</td>
<td>CORSBehavior</td>
<td>Configuration parameter for CORS behavior. Setting the value to 0 disables the CORS filter. Setting the value to 1 enables the CORS filter. Setting the value to 2 enables the CORS filter with the whitelist option.</td>
</tr>
</tbody>
</table>

### Manage External Authentication view

The **Manage External Authentication** view allows you to add, update, and delete an external authentication service. The external authentication service must first be implemented by a system administrator as described in *Authentication* on page 143. Once the service has been implemented, it can be added to a tenant.

The **Manage External Authentication** view is available to users with either set of the following permissions.

- Administrator (12) permission
- WebUI (8) permission, RegisterExternalAuthService (26) permission, and administrative access on the given tenant

The following table provides permissions and descriptions for actions in the **Manage External Authentication** view.
### System Configurations view

The **System Configurations** view can be used to set a number of configurations across the Hybrid Data Pipeline system. This view is only available to users with the Administrator (12) permission (system administrators).

<table>
<thead>
<tr>
<th>Action</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter roles by tenant</td>
<td>Administrative access to multiple tenants Web UI (8) RegisterExternalAuthService (26)</td>
<td>An administrator with administrative access to multiple tenants will have the option of selecting the tenant for which he or she wants to view or manage external authentication services. Select the tenant for which you want to view authentication services from the <strong>Select Tenant</strong> dropdown.</td>
</tr>
<tr>
<td>Register an external authentication service</td>
<td>Administrative access for the tenant Web UI (8) RegisterExternalAuthService (26)</td>
<td>To register an authentication service with the tenant, click <strong>+ New Service</strong>. Provide the following information, and then click <strong>Save</strong>.</td>
</tr>
</tbody>
</table>
|                                                  |                                                                            | - The name and description of the service  
|                                                  |                                                                            | - The service type  
|                                                  |                                                                            | - For Java plugin service provide:  
|                                                  |                                                                            |   - The class name  
|                                                  |                                                                            |   - Attributes  
|                                                  |                                                                            | - For LDAP service provide:  
|                                                  |                                                                            |   - Target URL  
|                                                  |                                                                            |   - Service Authentication  
|                                                  |                                                                            |   - Security Principal  
|                                                  |                                                                            |   - Other Attributes  
| Edit an external authentication service          | Administrative access for the tenant Web UI (8) RegisterExternalAuthService (26) | To edit an authentication service, select the service. Then, select **Edit** from the **Actions** dropdown. Edit the service as desired. |
| Delete an external authentication service        | Administrative access for the tenant Web UI (8) RegisterExternalAuthService (26) | To delete a service, select the service you want to delete. Then, select **Delete** from the **Actions** dropdown. Confirm or cancel the delete operation in the dialog. |

---

**Note:** Any user with Administrator (12) permission may perform all actions.
The following table provides descriptions of the options available via the **System Configurations** view.

<table>
<thead>
<tr>
<th>Option</th>
<th>Permissions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delimiter</td>
<td>Administrator (12)</td>
<td>Specifies a delimiter to be used between the user name and authentication service name. In the following example, the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>symbol delimits user437 and the LDAP1 service: user437</td>
</tr>
<tr>
<td>Secure Password Change</td>
<td>Administrator (12)</td>
<td>Specifies whether the current password is required in order to update the password of the logged-in user. The default value is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Default OData Version</td>
<td>Administrator (12)</td>
<td>Sets the default OData version for new data sources.</td>
</tr>
<tr>
<td>Default Entity Name</td>
<td>Administrator (12)</td>
<td>Sets the default entity name mode for OData V4 data sources. For details, see <a href="#">Configuring data sources for OData connectivity and working with data source groups</a> on page 622.</td>
</tr>
<tr>
<td>JDBC DataStore</td>
<td>Administrator (12)</td>
<td>Enables the third party JDBC data store feature. The default value is <strong>ON</strong>. For details, see <a href="#">Using third party JDBC drivers with Hybrid Data Pipeline</a> on page 182.</td>
</tr>
<tr>
<td>Password Policy</td>
<td>Administrator (12)</td>
<td>Enables the <strong>default password policy</strong>. The default value is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Option</td>
<td>Permissions</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>System Monitor Details</td>
<td>Administrator (12)</td>
<td>Determines how the system persists monitor details.</td>
</tr>
<tr>
<td>IP WhiteList Filtering</td>
<td>Administrator (12)</td>
<td>Enables the whitelist filtering feature. The default value is ON. See Implementing IP address whitelists on page 164 for details.</td>
</tr>
</tbody>
</table>

### Product information

Users can access product information by clicking the question mark icon and selecting **About**.

![About Hybrid Data Pipeline](image)

The **About Hybrid Data Pipeline** window displays installation and version information.

### User profile

The down arrow next to the username in the upper right hand corner of the Web UI opens a dropdown menu. Users can change their passwords by selecting the **Change Password** item, or log out by selecting the **Log Out** item.

### Changing your password in the Web UI

Take the following steps to change your password in the Web UI.
Note: You can also change your password using the Change Password API.

1. Select the arrow next to your username in the right hand corner of the Web UI.
2. Click Change Password to open the Change Password window.
3. Enter your current password in the Current Password field.
4. Enter your new password in the New Password field.

   Note: The password must be a maximum of 32 characters in length.

5. Retype your new password in the Confirm Password field.
6. Click SAVE.

Creating data sources with the Web UI

Hybrid Data Pipeline enables access to a variety of data stores, such as Apache Hive, DB2, SQL Server, Oracle, and Salesforce. To access data residing on a backend data store, Hybrid Data Pipeline administrators or users must create a Hybrid Data Pipeline data source. A Hybrid Data Pipeline data source can be created by specifying parameters associated with a specific data store. The information provided in the data source allows the service to connect to the backend data store. A data source can be created with the Web UI or the Data Sources API.

Note: While administrators can create their own data sources with the Web UI, they cannot create or modify data sources on behalf of users in the Web UI. In addition, administrators cannot set permissions on data sources with the Web UI. To create data sources on behalf of a user or set permissions on data sources, an administrator must execute API operations with the Data Sources API. See User provisioning on page 107 for use cases with example API operations.

Hybrid Data Pipeline also supports OData access to backend data stores. This access is enabled by specifying the appropriate parameters and configuring an OData schema under the OData tab. OData access occurs over HTTPS (or HTTP) and does not require a driver to be installed locally. Each OData-enabled data source exposes an OData schema. The name of this data source becomes part of the resource path in the OData URI. A data source group can be created to enable OData access from multiple schemas using a single resource path. A data source group can contain references to multiple data sources that have been enabled for OData. These data sources can be specified when the group is created, or added later. For more information, see Configuring data sources for OData connectivity and working with data source groups on page 622.

In addition, Hybrid Data Pipeline supports SQL read-only access to JSON-based REST services through the Autonomous REST Connector. When you create a REST data source, the connector creates a relational model of the returned JSON data and translates SQL statements to REST API requests. You can create and manage REST data sources either through the Web UI or through the Hybrid Data Pipeline API. For details, see Creating and using REST data sources on page 637 and Autonomous REST Connector parameters on page 257.

The following topics provide instructions on how to create a data source using the Web UI. The first topic provides step-by-step instructions for creating a data source. The subsequent topics describe the parameters that can be used to define a data source for each data store supported by Hybrid Data Pipeline.
How to create a data source in the Web UI

A Hybrid Data Pipeline data source contains the information that allows the service to connect to the backend data store.

Take the following steps to create a Hybrid Data Pipeline data source.

**Note:** These steps apply generally to all data stores, but the available options differ by data store type. Consult the topics that follow for information specific to supported data stores.

1. Navigate to the **Data Sources** view by clicking the data sources icon.
2. Click **+ NEW DATA SOURCE** to open the **Data Stores** page.
3. From the list of data stores, click data store to which you want to connect.
   
   The **Create Data Source** page opens.
4. Provide required information in the fields provided under each of the tabs.
5. Click **Save** to create the data source definition.
6. Click **TEST** to establish a connection with the data store.

If you create an OData-enabled data source, the icon beside it indicates the status of the schema map generation. The following table provides details of the icons:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>The synchronization of the schema map is in progress. The number denotes the percentage of synchronization completed.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>The schema map was synchronized successfully.</td>
</tr>
<tr>
<td><img src="image" alt="Icon" /></td>
<td>The schema map was synchronized successfully, but there are some table/column warnings. Hybrid Data Pipeline allows users to know the details of the tables/columns and/or functions that were dropped while generating the OData Model for a given schema map of a Data Source. The number of warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the <strong>Schema API</strong> on page 1412 to retrieve table and column warnings.</td>
</tr>
</tbody>
</table>
Errors occurred while synchronizing the schema map. You must address the errors and synchronize the schema map again. Hybrid Data Pipeline allows users to know the details of the tables and/or columns that were dropped while generating the OData Model for a given schema map of a Data Source. The number of errors/warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the Schema API on page 1412 to retrieve table and column warnings.

You must synchronize the schema map again.

**Supported data stores**

The parameters used to create a Hybrid Data Pipeline data source vary across supported data stores. See the topics listed in this table to review parameters specific to supported data stores.

**Note:**

- For connectivity using a third party JDBC driver, see Using third party JDBC drivers with Hybrid Data Pipeline on page 182 and JDBC parameters for third party drivers on page 292.
- For connectivity to REST services, see Creating and using REST data sources on page 637 and Autonomous REST Connector parameters on page 257.

<table>
<thead>
<tr>
<th>Data store</th>
<th>Supported Connection Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Redshift</td>
<td>Amazon Redshift parameters on page 227</td>
</tr>
<tr>
<td>Apache Hadoop Hive</td>
<td>Apache Hadoop Hive parameters on page 240</td>
</tr>
<tr>
<td>Autonomous REST Connector</td>
<td>Autonomous REST Connector parameters on page 257</td>
</tr>
<tr>
<td>DB2</td>
<td>DB2 parameters on page 272</td>
</tr>
<tr>
<td>JDBC third party</td>
<td>JDBC parameters for third party drivers on page 292</td>
</tr>
<tr>
<td>Google Analytics</td>
<td>Google Analytics parameters on page 298</td>
</tr>
<tr>
<td>Greenplum</td>
<td>Greenplum parameters on page 318</td>
</tr>
<tr>
<td>Informix</td>
<td>Informix parameters on page 334</td>
</tr>
<tr>
<td>Microsoft Dynamics CRM</td>
<td>Microsoft Dynamics CRM parameters on page 345</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>Microsoft SQL Server parameters on page 358</td>
</tr>
<tr>
<td>Data store</td>
<td>Supported Connection Parameters</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MySQL Community Edition</td>
<td>MySQL Community Edition parameters on page 384</td>
</tr>
<tr>
<td>MySQL Enterprise</td>
<td>MySQL Enterprise parameters on page 390</td>
</tr>
<tr>
<td>Oracle</td>
<td>Oracle parameters on page 405</td>
</tr>
<tr>
<td>Oracle Marketing Cloud (Eloqua)</td>
<td>Oracle Marketing Cloud (Eloqua) parameters on page 435</td>
</tr>
<tr>
<td>Oracle Sales Cloud</td>
<td>Oracle Sales Cloud parameters on page 449</td>
</tr>
<tr>
<td>Oracle Service Cloud</td>
<td>Oracle Service Cloud parameters on page 462</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>PostgreSQL parameters on page 476</td>
</tr>
<tr>
<td>Progress OpenEdge</td>
<td>Progress OpenEdge parameters on page 492</td>
</tr>
<tr>
<td>Progress Rollbase</td>
<td>Progress Rollbase parameters on page 506</td>
</tr>
<tr>
<td>Salesforce-based data sources</td>
<td>• Salesforce parameters on page 519</td>
</tr>
<tr>
<td></td>
<td>• FinancialForce parameters on page 538</td>
</tr>
<tr>
<td></td>
<td>• ServiceMax parameters on page 556</td>
</tr>
<tr>
<td></td>
<td>• Veeva CRM parameters on page 574</td>
</tr>
<tr>
<td>SugarCRM</td>
<td>SugarCRM parameters on page 592</td>
</tr>
<tr>
<td>Sybase ASE</td>
<td>Sybase parameters on page 606</td>
</tr>
</tbody>
</table>

**Amazon Redshift parameters**

The following tables describe parameters available on the tabs of an Amazon Redshift **Data Source** dialog:

- General tab
- Security tab
- OData tab
- Advanced tab
General tab

Create Amazon Redshift Data Source

- Data Source Name
- Description
- User ID
- Password
- Server Name
- Port Number
- Database

* Required Fields
### Table 2: General tab connection parameters for Amazon Redshift

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Source Name</strong></td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>A general description of the data source.</td>
</tr>
</tbody>
</table>
| **User Id**            | The login credentials for your Amazon Redshift server. Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the server must grant permission to a user with these credentials to access the data store and the target data.  
**Note:** You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline account. |
| **Password**           | A case-sensitive password that is used to connect to your Amazon Redshift database. A password is required if user ID/password authentication is enabled on your database. Contact your system administrator to obtain your password.  
**Note:** By default, the password is encrypted.  
By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password. |
| **Server Name**        | Specifies either the IP address in IPv4 or IPv6 format, or a combination of the two, or the server name (if your network supports named servers) of the primary database server, for example, RedshiftServer or 122.23.15.12  
**Valid Values:**  
`server_name | IP_address`  
where:  
`server_name`  
is the name of the server to which you want to connect.  
`IP_address`  
is the IP address of the server to which you want to connect. |
<p>| <strong>Port Number</strong>        | The port number of the Amazon Redshift server. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database*</td>
<td>The name of the database that is running on the database server.</td>
</tr>
</tbody>
</table>

**Security tab**

---

Table 3: Security tab Connection Parameters for Amazon Redshift

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crypto Protocol Version</td>
<td>Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>cryptographic_protocol [, cryptographic_protocol ]...</td>
</tr>
<tr>
<td></td>
<td>where:</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>cryptographic_protocol</td>
</tr>
<tr>
<td></td>
<td>is one of the following cryptographic protocols:</td>
</tr>
<tr>
<td></td>
<td>TLSv1</td>
</tr>
<tr>
<td></td>
<td>The client must send the highest version that it supports in the client hello.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.</td>
</tr>
<tr>
<td></td>
<td><strong>Example</strong></td>
</tr>
</tbody>
</table>
### Field: Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

**Default:** TLSv1, TLSv1.1, TLSv1.2

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TLSv1.1,TLSv1.2</td>
<td></td>
</tr>
</tbody>
</table>

### Field: Host Name In Certificate

Specifies a host name for certificate validation when validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.

**Valid Values:**

- `host_name` | `#SERVERNAME#`

where `host_name` is a valid host name.

If `host_name` is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the Hybrid Data Pipeline connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.

If `#SERVERNAME#` is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the Hybrid Data Pipeline connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.

**Default:** Empty string
**OData tab**

Create Amazon Redshift Data Source

Table 4: OData tab connection parameters for Amazon Redshift

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
### Field
- **OData Access URI**
  - Specifies the base URI for the OData feed to access the data source, for example, `https://example.com:8443/api/odata4/<datasourcename>`. You can copy the URI and paste it into your application's OData configuration.

  The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

  The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding `/$metadata` to the service root URI.

- **Schema Map**
  - Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.

  See [Configuring data sources for OData connectivity and working with data source groups](#) on page 622 for more information.

- **Data Source Caching**
  - Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

  Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

  When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

  **Valid Values:**

  When set to 1, session caching is enabled. This provides better performance for production.

  When set to 0, session caching is disabled. Use this value when you are configuring the data source.

  **Default:** 1
### Page Size

Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the `$top` and `$skip` parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.

**Valid Values:** $0 | n

where $n$ is an integer from 1 to 10000.

When set to 0, the server default of 2000 is used.

**Default:** 0

### Refresh Result

Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.

**Valid Values:**

When set to 0, the OData service caches the first page of results.

When set to 1, the OData service re-executes the query.

**Default:** 1

### Inline Count Mode

Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.

**Valid Values:**

When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.

When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Top Mode      | Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries. **Valid Values:**
  Set to **0** when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.
  Set to **1** when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.
  **Default:** 0                                                                                                                                                                                                                           |
| OData Read Only | Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.
  Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.
  **Valid Values:**
  **ON | OFF**
  When **ON** is selected, OData access is restricted to read-only mode.
  When **OFF** is selected, write operations can be performed on the OData service.
  **Default:** OFF                                                                                                                                                                                                                     |
Advanced tab

Create Amazon Redshift Data Source

Catalog Options

Extended Options

Initialization String

Login Timeout

Max Pooled Statements

Query Timeout

Result Set Meta Data Options

Metadata Exposed Schemes

* Required Fields
Table 5: Advanced tab connection parameters for Amazon Redshift

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Valid Values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog Options</td>
<td>Determines which type of metadata information is included in result sets when an application calls DatabaseMetaData methods. To include multiple types of metadata information, add the sum of the values that you want to include. In this case, specify 6 to query database catalogs for column information and to emulate getColumns() calls.</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If set to 2, the Hybrid Data Pipeline connectivity service queries database catalogs for column information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If set to 4, a hint is provided to the connectivity service to emulate getColumns() calls using the ResultSetMetaData object instead of querying database catalogs for column information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumns(). The argument to getColumns() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the Hybrid Data Pipeline connectivity service reverts to the default behavior for getColumns() calls.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Default: 2</td>
</tr>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]</td>
<td>If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid Values:</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td>Default: none</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Initialization String**

A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.

**Syntax:**

```plaintext
command[ ; command]...
```

Where:

- `command` is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:

```plaintext
InitializationString=(REFRESH SCHEMA SFORCE)
```

The default is an empty string.

**Login Timeout**

The amount of time, in seconds, that the Hybrid Data Pipeline connectivity service waits for a connection to be established before timing out the connection request.

**Valid Values:**

- `0` | `x`

  where `x` is a positive integer that represents a number of seconds.

  - If set to `0`, the connectivity service does not time out a connection request.
  - If set to `x`, the connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.

  **Default:** `30`

**Max Pooled Statements**

The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Query Timeout| Sets the default query timeout (in seconds) for all statements created by a connection. **Valid Values:** 

-1 | 0 | x

If set to -1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the Statement.setQueryTimeout() method.
If set to 0, the default query timeout is infinite (the query does not time out).
If set to x, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection option, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.

**Default:** 0
Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.

**Valid Values:**
0 | 1

If set to 0 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getTableName() method may return an empty string for each column in the result set.

If set to 1 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the ResultSetMetaData.getSchemaName() and ResultSetMetaData.getCatalogName() methods are called if the Hybrid Data Pipeline connectivity service can determine that information.

**Default:** 0

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

<schema>

Where:

<schema>

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

**See the steps for:**

How to create a data source in the Web UI on page 225

### Apache Hadoop Hive parameters

The following tables describe parameters available on the tabs of an Apache Hadoop Hive On-Premise Data Source dialog:
General tab

Table 6: General tab connection parameters for Apache Hadoop Hive

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
</tbody>
</table>
## Field and Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The port number of the Apache Hive server to connect to.</td>
</tr>
<tr>
<td>Password</td>
<td>A password for the Apache Hive account that is used to establish the connection to your Apache Hive server.</td>
</tr>
<tr>
<td><strong>Note:</strong> By default, the password is encrypted.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Server Name</td>
<td>Specifies either the server name (if your network supports named servers) or the IP address of the primary Apache Hive server machine, for example, MyHiveServer or 122.23.15.12.</td>
</tr>
<tr>
<td>User ID</td>
<td>The User ID for the Apache Hive account used to establish the connection to the Apache Hive server.</td>
</tr>
</tbody>
</table>
Security tab

Create Apache Hadoop Hive Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption Method</td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the database server.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>noEncryption</td>
</tr>
<tr>
<td></td>
<td>If set to noEncryption, data is not encrypted or decrypted.</td>
</tr>
<tr>
<td></td>
<td>If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the Login Timeout parameter to avoid problems when connecting to a server that does not support SSL.</td>
</tr>
<tr>
<td></td>
<td>• When SSL is enabled, the following parameters also apply:</td>
</tr>
<tr>
<td></td>
<td>Host Name In Certificate</td>
</tr>
<tr>
<td></td>
<td>Validate Server Certificate</td>
</tr>
<tr>
<td></td>
<td>Crypto Protocol Version</td>
</tr>
<tr>
<td>Default:</td>
<td>noEncryption</td>
</tr>
</tbody>
</table>
### Crypto Protocol Version

Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.

**Valid Values:**

```
cryptographic_protocol [, cryptographic_protocol ]...`
```

where:

```
cryptographic_protocol
```

is one of the following cryptographic protocols:

```
TLSv1 | TLSv1.1 | TLSv1.2
```

The client must send the highest version that it supports in the client hello.

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.

### Example

Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

```
TLSv1.1, TLSv1.2
```

**Default:** TLSv1, TLSv1.1, TLSv1.2
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Name In Certificate</td>
<td>Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.</td>
</tr>
</tbody>
</table>

**Valid Values:**

```
host_name | #SERVERNAME#
```

where `host_name` is a valid host name.

If `host_name` is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.

If `#SERVERNAME#` is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.

**Default:** Empty string
**Field** | **Description**
---|---
ImpersonateUser | Specifies the user ID used for Impersonation. When Impersonation is enabled on the server (`hive.server2.enable.doAs=true`), this value determines your identity and access rights to Hadoop resources when executing queries. If Impersonation is disabled, you will execute queries as the user who initiated the HiveServer2 process.

Validate Server Certificate | Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

**Valid Values:**

- **ON**
- **OFF**

If **ON** is selected, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file.

If **OFF** is selected, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any truststore information that is specified by the Java system properties.

**Default:** **ON**

**OData tab**

The following table describes the controls on the **OData tab**. For information on using the **Configure Schema** editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 8: OData tab connection parameters for Apache Hadoop Hive

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
**OData Access URI** | Specifies the base URI for the OData feed to access your data source, for example, https://hybridpipe.operations.com/api/odata/<DataSourceName>. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.

**Schema Map** | Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.

**Data Source Caching** | Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. **Valid Values:**
When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. **Default:** 1
### Page Size

Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the `$top` and `$skip` parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.

**Valid Values:** $0 | $n

where $n$ is an integer from 1 to 10000.

When set to 0, the server default of 2000 is used.

**Default:** 0

### Refresh Result

Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.

**Valid Values:**
- When set to 0, the OData service caches the first page of results.
- When set to 1, the OData service re-executes the query.

**Default:** 1

### Inline Count Mode

Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.

**Valid Values:**
- When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.
- When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

**Default:** 1
### Field: Top Mode

Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**
- Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.
- Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.

**Default:** 0

### Field: OData Read Only

Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.

Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.

**Valid Values:**
- ON | OFF
  - When ON is selected, OData access is restricted to read-only mode.
  - When OFF is selected, write operations can be performed on the OData service.

**Default:** OFF

### Field: String Max Length

Controls the maximum length reported for Apache Hive String columns. Values larger than the specified value cause the String columns to be excluded from the model. Values smaller than the specified value may cause issues with some OData applications as data may be returned that exceeds the maximum length. The default value is 32768.
Advanced tab

Create Apache Hadoop Hive Data Source

- Array Fetch Size: 20000
- Array Insert Size: 20000
- Batch Mechanism: multiRowInsert
- Catalog Mode: mixed
- Initialization String
- Login Timeout: 30
- Max Pooled Statements: 0
- Query Timeout: 0
- Transport Mode: binary
- HTTP Path: diservice
- Enable Cookie Authentication: ON
- Cookie Name: hive.server2.auth
- Extended Options
- Metadata Exposed Schemas

[Image of the Advanced tab section]
### Table 9: Advanced tab connection parameters for Apache Hadoop Hive

| Field               | Description                                                                                                                                                                                                reds
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Array Fetch Size    | Specifies the number of fields the data access service uses to calculate the maximum number of rows for a fetch. When executing a fetch, the service divides the Array Fetch Size value by the number of columns in a particular table to determine the number of rows to retrieve. By determining the fetch size based on the number of fields, out of memory errors may be avoided when fetching from tables containing a large number of columns while continuing to provide improved performance when fetching from tables containing a small number of columns. Valid values: 

\[-x \mid x\]

where:

\[-x\] is a negative integer

\[x\] is a positive integer.

If set to \(-x\), the service overrides any settings on the statement level and uses the number of fields specified by the absolute value of \(-x\) to calculate the number of rows to retrieve.

If set to \(x\), the service uses the number of fields specified by the value of \(x\) to calculate the number of rows to retrieve. However, the service will not override settings, such as setFetchSize(), on the statement level.

For example, if this property is set to 20000 fields and you are querying a table with 19 columns, the service divides the number of fields by the number of columns to calculate the number of rows to retrieve. In this case, approximately 1053 rows would be retrieved for each fetch.

**Note:** You can improve performance by increasing the value specified for this parameter. However, if the number of fields specified exceeds the available buffer memory on the server, an out of memory error will be returned. If you receive this error, decrease the value specified until fetches are successfully executed.

**Default:** 20000 (fields)
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Array Insert Size     | Specifies the number of fields the data access service uses to calculate the maximum number of rows sent in a packet when executing a multi-row insert. When executing a multi-row insert, the service divides the Array Insert Size value by the number of columns in a particular insert statement to determine the number of rows to send in a packet. By determining the packet size based on the number of fields, the service can avoid out of memory errors when executing inserts containing a large number of columns while continuing to provide improved performance when executing inserts containing a small number of columns. The default value is 20,000 fields. In most scenarios, the default setting for Array Insert Size provides the ideal behavior; however, you may need to reduce the value specified if you encounter either of the following:  
  • Performance or memory issues when inserting a large number of rows that contain large values.  
  • The following error when inserting a large number of rows when using Apache Knox: HTTP/1.1 500 Server Error.  
  **Default:** 20000 (fields) |
| Batch Mechanism       | Determines the mechanism that is used to execute batch operations.  
  **Valid values:**  
  nativeBatch | multiRowInsert.  
  If set to nativeBatch, the Hive native batch mechanism is used to execute batch operations, and an insert statement is executed for each row contained in a parameter array.  
  If set to multiRowInsert, the service attempts to execute a single insert statement for all the rows contained in a parameter array. If the size of the insert statement exceeds the available buffer memory of the server, the service executes multiple statements. This behavior provides substantial performance gains for batch inserts.  
  **Default:** multiRowInsert |

**Note:**  
• Multirow inserts can only be performed on Insert statements that use parameterized arrays.  
• Batch operations for parameterized arrays are not supported for updates or deletes.  
• The service modifies the HQL statement to perform a multirow insert.  
• This connection property can affect performance.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog Mode</td>
<td>Specifies whether the service uses native catalog functions to retrieve information returned by DatabaseMetaData functions.</td>
</tr>
<tr>
<td><strong>Valid values:</strong></td>
<td>mixed</td>
</tr>
<tr>
<td></td>
<td>If set to <strong>mixed</strong>, the service uses a combination of native catalog functions and discovered information to retrieve catalog information. Select this option for the optimal balance of performance and accuracy.</td>
</tr>
<tr>
<td></td>
<td>If set to <strong>native</strong>, the service uses native catalog functions to retrieve information returned by DatabaseMetaData functions. This setting provides the best performance, but at the expense of less-accurate catalog information.</td>
</tr>
<tr>
<td></td>
<td>If set to <strong>query</strong>, the service uses discovered information to retrieve catalog information. This option provides highly accurate catalog information, but at the expense of slower performance.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>mixed</td>
</tr>
<tr>
<td>Initialization String</td>
<td>A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.</td>
</tr>
<tr>
<td><strong>Syntax:</strong></td>
<td>command[[; command]...]</td>
</tr>
<tr>
<td><strong>Where:</strong></td>
<td>command</td>
</tr>
<tr>
<td></td>
<td>is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:</td>
</tr>
<tr>
<td></td>
<td>InitializationString=(REFRESH SCHEMA SFORCE)</td>
</tr>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, that the Hybrid Data Pipeline connectivity service waits for a connection to be established before timing out the connection request.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>where x is a positive integer that represents a number of seconds.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service does not time out a connection request.</td>
</tr>
<tr>
<td></td>
<td>If set to x, the connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>30</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.</td>
</tr>
<tr>
<td>Query Timeout</td>
<td>The number of seconds for the default query timeout for all statements that are created by a connection.</td>
</tr>
<tr>
<td>Valid Values:</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>If set to −1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the Statement.setQueryTimeout() method.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the default query timeout is infinite (the query does not time out).</td>
</tr>
<tr>
<td></td>
<td>If set to x, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection parameter, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>Transport Mode</td>
<td>Specifies whether binary (TCP) mode or HTTP mode is used to access Apache Hive data sources.</td>
</tr>
<tr>
<td>Valid values:</td>
<td>binary</td>
</tr>
<tr>
<td></td>
<td>If set to binary, Thrift RPC requests are sent directly to data sources using a binary connection (TCP mode).</td>
</tr>
<tr>
<td></td>
<td>If set to http, Thrift RPC requests are sent using HTTP transport (HTTP mode). HTTP mode is typically used when connecting to a proxy server, such as a gateway, for improved security, or a load balancer.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> binary</td>
</tr>
</tbody>
</table>

**Note:**
- The setting of this parameter corresponds to that of the hive.server2.transport.mode property in your hive-site.xml file.
- When Transport Mode is set to http, the HTTP/HTTPS end point for the Hive server must be specified using the HTTP Path parameter.
- To use HTTPS end points, set Transport Mode to http and Encryption Method to SSL.
- Apache Hive currently supports using only one protocol mode per server at a time.
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **HTTP Path**                 | Specifies the path of the HTTP/HTTPS endpoint used for connections when HTTP mode is enabled (Transport Mode set to http). \n
**Valid values:**  \n
*string*  \n
where:  \n
*string*  \n
is the path of the URL endpoint. By default, the value specified must be an HTTP endpoint. To support HTTPS values, enable SSL by setting Encryption Method to SSL. |
| **Enable Cookie Authentication** | Determines whether the service attempts to use cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. Cookie based authentication improves response time by eliminating the need to re-authenticate with the server for each request.  \n
**Valid values:**  \n
ON | OFF  \n
If set to ON, the service attempts to use cookie based authentication for requests to an HTTP endpoint after the initial authentication to the server. The cookie used for authentication is specified by the **Cookie Name** parameter. If the name does not match, or authentication fails, the driver attempts to authenticate according to the setting of the **Authentication Method**.  \n
If set to OFF, the service does not use cookie based authentication for HTTP requests after the initial authentication.  \n
**Default:** ON |
| **Cookie Name**               | Specifies the name of the cookie used for authenticating HTTP requests when HTTP mode is enabled (Transport Mode set to http) and cookie based authentication is enabled (Enable Cookie Authentication is set to ON). When preparing an HTTP request to the server, the service will not attempt to reauthenticate if a valid cookie is present.  \n
**Valid values:**  \n
*string*  \n
where:  \n
*string*  \n
is a valid cookie name.  \n
**Default:** hive.server2.auth |
### Extended Options

Specifies a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.

**Valid Values:**

- `string`

**Default:** none

### Metadata Exposed Schemas

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

- `<schema>`

**Where:**

- `<schema>`

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

### Autonomous REST Connector parameters

**Note:** For additional information about REST connectivity, see Creating and using REST data sources on page 637.

The following tables describe parameters available on the tabs of an Autonomous REST Connector Data Source setup dialog:
Chapter 3: Using Hybrid Data Pipeline

- General tab
- Security tab
- OData tab
- Mapping tab
- Advanced tab

**General tab**

Create Autonomous REST Connector Data Source

<table>
<thead>
<tr>
<th>General</th>
<th>Security</th>
<th>OData</th>
<th>Mapping</th>
<th>Advanced</th>
</tr>
</thead>
</table>

**Data Source Name**

*Required Fields*

**Description**

**Endpoints**

- Add
- Import REST Input

- Generate Configuration

**Connector ID**
Table 10: General tab connection parameters

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Endpoints</td>
<td>Specify REST endpoints in either of the following ways. The REST endpoints specified are used to generate a relational model of the REST data.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Option 1.</strong> Add REST endpoints via the Web UI. Click <strong>Add</strong>, and provide the following information in the dialog.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Entity Name</strong> is the name of the relational table to which the connectivity service maps the endpoint.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Request Type</strong> is the type of request that is used to retrieve data from the endpoint. (If <strong>POST</strong> is selected, a <strong>HTTP Body</strong> field will be provided.)</td>
</tr>
<tr>
<td></td>
<td>• <strong>URL</strong> is the URL of the REST endpoint. For example, <a href="http://mysite.com/countries/">http://mysite.com/countries/</a>.</td>
</tr>
<tr>
<td></td>
<td>• <strong>JSON Root</strong> is the JSON object which the connectivity service uses to develop a tabular data model. When a JSON root is specified, JSON layers between the endpoint and the root are ignored.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Option 2.</strong> Import an input REST file. Click <strong>Import REST file</strong>, and browse to the input REST file you want to import. For information on creating an input REST file using a text editor, see <strong>Creating an input REST file</strong> on page 641.</td>
</tr>
</tbody>
</table>

**Take the following steps to refine the relational model of REST data.**

1. Click the generate (or edit) configuration button.
2. Edit the JSON to meet application or query requirements. See **Creating an input REST file** on page 641 for syntax requirements.
3. Click **Update** in the editor to save your changes.
4. Click **Update** in the data source dialog to update the data source.

<table>
<thead>
<tr>
<th>Connector ID</th>
<th>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty.</td>
</tr>
<tr>
<td></td>
<td>If you own multiple Connectors that have the same name, for example, <strong>Production</strong>, an identifier is appended to each Connector, for example, <strong>Production_dup0</strong> and <strong>Production_dup1</strong>. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, <strong>Production(owner1)</strong> and <strong>Production(owner2)</strong>.</td>
</tr>
</tbody>
</table>
Security tab

Create Autonomous Rest Connector Data Source

<table>
<thead>
<tr>
<th>Authentication Method</th>
<th>Required Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

User
Password
Authentication HTTP Header Name
Authentication URL Param Name
Security Token

Table 11: Security tab connection parameters

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Method</td>
<td>Determines which authentication method the connectivity service uses during</td>
</tr>
<tr>
<td></td>
<td>the course of a session.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>Basic</td>
</tr>
<tr>
<td>When set to None, the service does not attempt to authenticate.</td>
<td></td>
</tr>
<tr>
<td>When set to Basic, the service uses a hashed value, based on the concatenation of the user name and password, for authentication. In addition to the User and Password properties, you must also configure the AuthHeader property if the name of your HTTP header is not Authorization (the default).</td>
<td></td>
</tr>
<tr>
<td>When set to HttpHeaders, the service passes security tokens via HTTP headers for authentication. You must also configure the SecurityToken property and, if the name of your HTTP header is not Authorization (the default), the AuthHeader property.</td>
<td></td>
</tr>
<tr>
<td>When set to UrlParameter, the service passes security tokens via the URL for authentication. You must also configure the AuthParam and SecurityToken properties.</td>
<td></td>
</tr>
<tr>
<td><strong>Default:</strong> None</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| User                          | Specifies the user name that is used to connect to the REST service. A user name is required if user is enabled by your REST service. This parameter is ignored when **Authentication Method** is set to **None**.  
**Valid Values:**  
`string`  
where:  
`string`  
is a valid user name. The user name is case-insensitive. |
| Password                      | Specifies the password to use to connect to your REST service. This parameter is ignored when **Authentication Method** is set to **None**.  
**Valid Values:**  
`password`  
where:  
`password`  
is a valid password. The password is case-sensitive. |
| Authentication HTTP Header Name | Specifies the name of the HTTP header used for authentication. This parameter is used when **Authentication Method** is set to **Basic** or **HttpHeader** authentication has been selected.  
**Valid Values:**  
`auth_header`  
where:  
`auth_header`  
is the name of the HTTP header used for authentication. For example, X-Api-Key. |
### OData tab

The following table describes the controls on the **OData** tab. For information on using the **Configure Schema** editor, see [Configuring data sources for OData connectivity and working with data source groups](#) on page 622. For information on formulating OData requests, see "Formulating queries" under [Querying with OData](#).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Authentication URL Param Name | Specifies the name of the URL parameter used to pass the security token. This property is required when **Authentication Method** is set to **UrlParameter**.  
**Valid Values:**  
*auth_parameter*  
where:  
*auth_parameter*  
is the name of the URL parameter used to pass the security token. For example, *apikey* or *key*. |
| Security Token               | Specifies the security token required to make a connection to your REST API endpoint. This parameter is required when **Authentication Method** is set to **UrlParameter** or **HttpHeader**. If a security token is required and you do not supply one, the connection will fail.  
**Important:** The **Security Token** parameter, like all parameters, is persisted in clear text.  
**Valid Values:**  
*string*  
where:  
*string*  
is the value of the security token assigned to the user. |
Table 12: OData tab connection parameters

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access the data source, for example, <a href="https://example.com:8443/api/odata4/">https://example.com:8443/api/odata4/</a>&lt;datasourcename&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. Valid Values: When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. Default: 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Page Size   | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** 0 | n  
where n is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0 |
| Refresh Result | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1 |
Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.

Valid Values:

When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.

When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

Default: 1

Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.

Valid Values:

Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.

Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.

Default: 0

The Mapping tab provides options for managing the relational map of the REST data.
Creating data sources with the Web UI
### Table 13: Mapping tab connection parameters

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh Schema</td>
<td>Specifies whether the connectivity service automatically refreshes the relational map of the data model when a user connects to a REST service.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td></td>
</tr>
<tr>
<td>ON</td>
<td>When set to ON, the service automatically refreshes the map of the data model when a user connects to a REST service. Changes to objects since the last time the map was generated will be shown in the metadata.</td>
</tr>
<tr>
<td>OFF</td>
<td>When set to OFF, the service does not refresh the map of the data model when a user connects to a REST service.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td></td>
</tr>
<tr>
<td>•</td>
<td>This parameter should not be enabled when Create Mapping is set to session.</td>
</tr>
<tr>
<td>•</td>
<td>You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.</td>
</tr>
<tr>
<td>•</td>
<td>Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.</td>
</tr>
<tr>
<td>•</td>
<td>If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>OFF</td>
</tr>
</tbody>
</table>

| Create Mapping      | Determines whether the connectivity service creates the internal files required for a relational map of the REST data when establishing a connection.                                                               |
| **Valid Values:**   |                                                                                                                                                                                                             |
| session             | When set to session, the service uses memory to store the internal configuration information and relational map of REST data. A REST file is not created when this value is specified. After the session, the view is discarded. |
| forceNew            | When set to forceNew, the service generates a new REST file and creates a new relational map of the REST data.                                                                                               |
| notExist            | **Warning:**  This causes all customizations defined in the REST file to be lost.                                                                                                                            |
|                     | When set to notExist, the service uses the current REST file and relational map of REST data. If the files do not exist, the service creates them.                                                            |
| **Default:**        | notExist                                                                                                                                                                                                  |
### Advanced tab

#### Create Autonomous REST Connector Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Call Limit</td>
<td>The maximum number of Web service calls allowed for a single SQL statement or metadata query. When set to 0, there is no limit on the number of Web service calls on a single connection that can be made when executing a SQL statement.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 10</td>
</tr>
</tbody>
</table>

**Table 14: Advanced tab connection parameters**
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Fetch Size</td>
<td>Specifies the number of rows of data the Hybrid Data Pipeline connectivity service attempts to fetch for each call.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>0</td>
</tr>
<tr>
<td>If set to 0, the Hybrid Data Pipeline connectivity service attempts to fetch up to a maximum of 10000 rows. This value typically provides the maximum throughput.</td>
<td></td>
</tr>
<tr>
<td>If set to x, the Hybrid Data Pipeline connectivity service attempts to fetch up to a maximum of the specified number of rows. Setting the value lower than 10000 can reduce the response time for returning the initial data. Consider using a smaller value for interactive applications only.</td>
<td></td>
</tr>
<tr>
<td><strong>Default:</strong> 0</td>
<td></td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 3.</td>
</tr>
<tr>
<td>Web Service Timeout</td>
<td>The time, in seconds, to wait before retrying a timed-out Select request. Valid only if the value of Web Service Retry Count is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.</td>
</tr>
<tr>
<td></td>
<td>The default is an empty string.</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td><em>string</em></td>
</tr>
<tr>
<td></td>
<td>where:</td>
</tr>
<tr>
<td></td>
<td><em>string</em></td>
</tr>
<tr>
<td></td>
<td>is a semi-colon separated list of connection options and their values.</td>
</tr>
<tr>
<td></td>
<td><strong>Syntax:</strong></td>
</tr>
<tr>
<td></td>
<td>Database=Server1;UndocumentedOption1=value1;UndocumentedOption2=value2;</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.</td>
</tr>
<tr>
<td>Metadata Exposed Schemas</td>
<td>Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.</td>
</tr>
<tr>
<td></td>
<td><strong>Warning:</strong> This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values</strong></td>
</tr>
<tr>
<td></td>
<td><em>&lt;schema&gt;</em></td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td><em>&lt;schema&gt;</em></td>
</tr>
<tr>
<td></td>
<td>is the name of a valid schema on the backend data store.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> No schema is specified. Therefore, all schemas are exposed.</td>
</tr>
</tbody>
</table>

**See the steps for:**

How to create a data source in the Web UI on page 225
DB2 parameters

Before you start
For the driver to create and bind packages with your user ID, these privileges are required: BINDADD for binding packages, CREATEIN for the collection specified by the Package Collection option, and GRANT EXECUTE for the PUBLIC group for executing the packages. Typically, a Database Administrator (DBA) has these privileges. If your user ID does not have these privileges, someone that has a user ID with DBA privileges must create packages by connecting to the connectivity service.

When connecting for the first time, the connectivity service determines whether bind packages exist on the server. If packages do not exist, the service creates them using the default values.

The following basic information enables you to connect with your data source and test your connection after installation.

- General tab
- OData tab
- Security tab
- Advanced tab

Parameters for a basic connection
The following table describes the connection parameters that you must supply on the **General** tab.

### Table 15: General tab connection parameters for DB2

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premises Connector that is used to access the on-premise data source. Click the arrow and select the Connector that you want to use. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premises Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, <code>Production</code>, an identifier is appended to each Connector, for example, <code>Production_dup0</code> and <code>Production_dup1</code>. If the Connectors in the drop-down list were shared with you, the owner's name is appended, for example, <code>Production(owner1)</code> and <code>Production(owner2)</code>.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of this set of connection parameters.</td>
</tr>
</tbody>
</table>
| Location Name       | Specifies the name of the DB2 location that you want to access. For DB2 for z/OS, your system administrator can determine the name of your DB2 location using the following command:  
                        | DISPLAY DDF                                                                                                                                                                                                    |
|                     | For DB2 for iOS, your system administrator can determine the name of your DB2 location using the following command. The name of the database that is listed as "LOCAL" is the value that you should use for this attribute.  
<pre><code>                    | WRKRDBDIRE                                                                                                                                                                                                     |
</code></pre>
<p>|                     | This option is not supported for DB2 for Linux/UNIX/Windows.                                                                                                                                                  |
|                     | This option is mutually exclusive with the Database Name option.                                                                                                                                              |
|                     | Valid Value:                                                                                                                                                                                                  |
|                     | location_name                                                                                                                                                                                               |
|                     | where:                                                                                                                                                                                                       |
|                     | location_name                                                                                                                                                                                               |
|                     | is the name of a valid DB2 location.                                                                                                                                                                          |
| Password            | A password for the DB2 account that is used to establish the connection to your DB2 server.                                                                                                                                 |
|                     | <strong>Note:</strong> The password for the DB2 account is different from your Hybrid Data Pipeline password.                                                                                                             |
|                     | <strong>Note:</strong> By default, the password is encrypted.                                                                                                                                                              |
|                     | By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye [ ] icon. Click the icon again to conceal the password.              |
| Port Number         | The TCP port of the primary database server that is listening for connections to the DB2 database.                                                                                                             |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Server Name   | Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, 122.23.15.12 or AppServer2.  
**Valid Values:**  
string  
where:  
string  
is the IP address or the name of the server to which you want to connect. |
| User Id       | The User ID for the DB2 account used to establish the connection to the DB2 server.  
**Note:** The User ID for the DB2 account is different from your Hybrid Data Pipeline User ID. |

**Security tab**

Create DB2 Data Source

[Image of Security tab interface]

- **Authentication Method**
- **Encryption Method**
- **Crypto Protocol Version**
- **Host Name In Certificate**
- **Validate Server Certificate**

[Image of security settings with buttons: Cancel, TEST, Save]
### Table 16: Security tab connection parameters for DB2

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Authentication    | Determines which authentication method the Hybrid Data Pipeline connectivity service uses when it establishes a connection. When user ID/password authentication is used, the encryption method that is used for user IDs and passwords is negotiated during the connection process. Supported encryption methods are:  
  • Advanced Encryption Standard (AES)  
  • Data Encryption Standard (DES)  
  To use AES encryption, the following requirements and restrictions apply:  
  • AES is supported for the following DB2 databases:  
    • DB2 V9.x and higher for Linux/UNIX/Windows  
    • DB2 UDB V8.1 for Linux/UNIX/Windows (requires DB2 Fix Pack 16)  
    • DB2 V9.1 for z/OS  
    • DB2 UDB V8.1 for z/OS (requires DB2 PTF for APAR PK56287)  
    • The Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy files, which require Java SE 5 or higher, must be installed with the On-Premise Connector on the client or application server. You can obtain these files from the following URL: [http://www.oracle.com/technetwork/java/javase/downloads/index.html](http://www.oracle.com/technetwork/java/javase/downloads/index.html)  
  • The DB2 authentication parameter on the database server must be set to a value of SERVER_ENCRYPT.  
  • For DB2 V9.7 for Linux/UNIX/Windows, the DB2 alternate_auth_enc parameter on the database server must be set to allow AES encryption.  
  • AES encryption cannot be used if the Encryption Method parameter is set to a value of DBEncryption or requestDBEncryption.  
| Valid Values      | clearText | encryptedPassword | encryptedPasswordAES | encryptedUIDPassword | encryptedUIDPasswordAES |

If set to clearText, the Hybrid Data Pipeline connectivity service uses user ID/password authentication. The connectivity service sends the user ID and password in clear text to the DB2 server for authentication. If a user ID and password are not specified, the connectivity service throws an exception.

If set to encryptedPassword, the Hybrid Data Pipeline connectivity service uses user ID/password authentication. The connectivity service sends a user ID in clear text and an encrypted password to the DB2 server for authentication. If the requirements for AES encryption are met, the connectivity service uses AES encryption; otherwise, the connectivity service allows a downgrade to DES encryption. If the Encryption Method parameter is set to a value of DBEncryption or requestDBEncryption, the Hybrid Data Pipeline connectivity service downgrades encryption to DES. If a user ID and password are not specified, the connectivity service throws an exception.
If set to `encryptedPasswordAES`, the Hybrid Data Pipeline connectivity service uses user ID/password authentication. The connectivity service sends a clear text user ID and an AES-encrypted password to the DB2 server for authentication. The Hybrid Data Pipeline connectivity service throws an exception in the following cases:

- If the database server indicates encryption must be downgraded to DES
- If a user ID and password are not specified
- If the Encryption Method parameter is set to a value of `DBEncryption` or `requestDBEncryption`

If set to `encryptedUIDPassword`, the Hybrid Data Pipeline connectivity service uses user ID/password authentication. The connectivity service sends an encrypted user ID and password to the DB2 server for authentication. If the requirements for AES encryption are met, the connectivity service uses AES encryption; otherwise, the connectivity service allows a downgrade to DES encryption. If the Encryption Method parameter is set to a value of `DBEncryption` or `requestDBEncryption`, the connectivity service downgrades encryption to DES. If a user ID and password are not specified, the connectivity service throws an exception.

If set to `encryptedUIDPasswordAES`, the Hybrid Data Pipeline connectivity service uses user ID/password authentication. The connectivity service sends an AES-encrypted user ID and password to the DB2 server for authentication. The connectivity service throws an exception in the following situations:

- If the database server indicates encryption must be downgraded to DES
- If a user ID and password are not specified
- If the Encryption Method parameter is set to a value of `DBEncryption` or `requestDBEncryption`.

**Note:**

- The User parameter provides the user ID. The Password parameter provides the password. The Encryption Method parameter determines whether the Hybrid Data Pipeline connectivity service uses data encryption.

**Default:** `clearText`

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Crypto Protocol Version**  | Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error. **Valid Values:**  
  
  `cryptographic_protocol [[, cryptographic_protocol ]...]
  
  where:  
  
  `cryptographic_protocol`
  
  is one of the following cryptographic protocols:  
  
  TLSv1 | TLSv1.1 | TLSv1.2 |
The client must send the highest version that it supports in the client hello.

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.

**Example**

Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

TLSv1.1, TLSv1.2

**Default:** TLSv1, TLSv1.1, TLSv1.2

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption Method</td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>noEncryption</td>
</tr>
</tbody>
</table>

If set to noEncryption, data is not encrypted or decrypted.

If set to DBEncryption, data is encrypted using DES encryption if the database server supports it. If the database server does not support DES encryption, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception. The Authentication Method parameter must be set to a value of clearText, encryptedPassword, or encryptedUIDPassword. This value is not supported for DB2 for i.

If set to requestDBEncryption, data is encrypted using DES encryption if the database server supports it. If the database server does not support DES encryption, the Hybrid Data Pipeline connectivity service attempts to establish an unencrypted connection. The Authentication Method parameter must be set to a value of clearText, encryptedPassword, or encryptedUIDPassword. This value is not supported for DB2 for i.

If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.

**Note:**

- Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You might want to set a login timeout using the Login Timeout property to avoid problems when connecting to a server that does not support SSL.
- When SSL is enabled, the following properties also apply:

  - Host Name In Certificate
  - ValidateServerCertificate
  - Crypto Protocol Version
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The default value is <code>noEncryption</code>.</td>
</tr>
</tbody>
</table>
### Field: Host Name In Certificate

Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.

**Valid Values:**

- `host_name` | `#SERVERNAME#`

where `host_name` is a valid host name.

If `host_name` is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the Hybrid Data Pipeline connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.

If `#SERVERNAME#` is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.

**Default:** Empty string

### Field: Validate Server Certificate
Determination whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

**Valid Values:**

- ON | OFF

If ON is selected, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate parameter is specified, the connectivity service also validates the certificate using a host name. The Host Name In Certificate parameter is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the connectivity service is connecting to is the server that was requested.

If OFF is selected, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any Java system properties.

**Default:** OFF

---

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 17: OData tab connection parameters for DB2

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Page Size   | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** $0 | n  
where $n$ is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0  |
| Refresh Result | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1  |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
**Valid Values:**  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
**Default:** 1  |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>**ON</td>
</tr>
<tr>
<td></td>
<td>When ON is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When OFF is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
</tbody>
</table>
### Advanced tab

#### Create DB2 Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alternate ID</td>
<td>Specifies the name of the schema to be used to qualify unqualified database objects in dynamically prepared SQL statements. This property sets the name of the schema in the DB2 CURRENT SCHEMA special register. If the attempt to change the schema fails, the connection fails and you receive the message <code>Invalid value for AlternateID</code>. Refer to your DB2 documentation for permission requirements imposed by the database.</td>
</tr>
</tbody>
</table>

**Valid Values:**
### Field: Alternate Servers

Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property.

**Valid Values:**

```plaintext
(servername1[:port1], servername2[:port2], ...) 
```

The server name (`servername1, servername2, and so on`) is required for each alternate server entry. Port number (`port1, port2, and so on`) is optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number is used.

**Default:** None

### Field: Catalog Options

Determines which type of metadata information is included in result sets when a JDBC application calls DatabaseMetaData methods. To include multiple types of metadata information, add the sum of the values that you want to include. In this case, specify 8 to include synonyms and to emulate `getColumns()` calls.

**Valid Values:**

```plaintext
0 | 2 | 6 
```

- If set to 0, result sets do not contain synonyms or remarks.
- If set to 2, result sets contain synonyms and remarks that are returned from the following DatabaseMetaData methods: `getColumns()`, `getExportedKeys()`, `getFunctionColumns()`, `getFunctions()`, `getImportedKeys()`, `getIndexInfo()`, `getPrimaryKeys()`, `getProcedureColumns()`, and `getProcedures()`.
- If set to 6, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate `getColumns()` calls using the ResultSetMetaData object instead of querying database catalogs for column information. Result sets contain synonyms, but not remarks. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using `getColumns()`. The argument to `getColumns()` must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the Hybrid Data Pipeline connectivity service reverts to the default behavior for `getColumns()` calls.

**Default:** 2
### Code Page Override

The code page to be used by the Hybrid Data Pipeline connectivity service to convert Character and Clob data. The specified code page overrides the default database code page or column collation. All Character and Clob data that is returned from or written to the database is converted using the specified code page.

By default, the Hybrid Data Pipeline connectivity service automatically determines which code page to use to convert Character data. Use this parameter only if you need to change the Hybrid Data Pipeline connectivity service’s default behavior.

**Valid Values:**

```string```

where `string` is the name of a valid code page that is supported by your JVM. For example, `CP950`.

**Default:** empty string

### Concurrent Access Resolution

Determines whether a read transaction can access committed rows that are locked by a write transaction when the application isolation level is Read Committed (DB2 Cursor Stability) or Repeatable Read (DB2 Read Stability). This parameter only applies to connections to DB2 V9.7 for Linux/UNIX/Windows and higher databases.

**Valid Values:**

```auto | useCurrentlyCommitted | waitForOutcome```

If set to `auto`, the connectivity service determines whether read transactions can access currently committed data when lock contention occurs by checking the setting of the DB2 `cur_commit` parameter on the database server. If the `cur_commit` parameter is set to `ON`, read transactions can access currently committed data.

If set to `useCurrentlyCommitted`, the connectivity service allows read transactions to access currently committed data if the data is being updated or deleted. Read transactions skip rows that are being inserted.

If set to `waitForOutcome`, read transactions wait for a commit or rollback operation if they encounter data that is being updated or deleted. Read transactions do not skip rows that are being inserted.

**Default:** auto

### Extended Options

Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.

**Valid Values:** `string`

**Default:** empty string
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization String</td>
<td>A semicolon delimited set of commands to be executed after the Hybrid Data Pipeline connectivity service has established and performed all initialization for the connection with DB2. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed. Syntax: command[[; command]...] Where: command is a SQL command. Multiple commands must be separated by semicolons. The following example for DB2 for z/OS adds USER2 to the CURRENT PATH special register and sets the CURRENT PRECISION special register to DEC31: SET CURRENT PATH=current_path, USER2;SET CURRENT PRECISION='DEC31' The default is an empty string.</td>
</tr>
<tr>
<td>Load Balancing</td>
<td>Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property. Valid Values: ON</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, to wait for a connection to be established before timing out the connection request.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>where x is a positive integer that represents a number of seconds.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service does not time out a connection request.</td>
</tr>
<tr>
<td></td>
<td>If set to x, the connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 30</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.</td>
</tr>
<tr>
<td>Query Timeout</td>
<td>Sets the default query timeout (in seconds) for all statements that are created by a connection.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>where x is a positive integer that represents a number of seconds.</td>
</tr>
<tr>
<td></td>
<td>If set to -1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the Statement.setQueryTimeout() method.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the default query timeout is infinite (the query does not time out).</td>
</tr>
<tr>
<td></td>
<td>If set to x, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value that is set by this parameter, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Result Set Meta Data Options | Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.  

**Valid Values:**

\[ 0 \mid 1 \]

If set to 0 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getTableName() method may return an empty string for each column in the result set.

If set to 1 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the ResultSetMetaData.getSchemaName() and ResultSetMetaData.getCatalogName() methods are called if the connectivity service can determine that information.  

**Default:** 0
### Metadata Exposed Schemas

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

`<schema>`

Where:

`<schema>`

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

### With Hold Cursors

Determines whether the cursor stays open on commit. After a commit, DB2 can leave all cursors open (Preserve cursors) or close all open cursors (Delete cursors). Rolling back a transaction closes all cursors regardless of how this property is specified.

**ON | OFF**

If set to **ON**, the cursor behavior is Preserve.

If set to **OFF**, the cursor behavior is Delete.

**Default:** **ON**

---

**JDBC parameters for third party drivers**

**Important:** Before you can proceed with creating a third party driver data source, an administrator must integrate the third party driver with Hybrid Data Pipeline. For detailed information, see [Using third party JDBC drivers with Hybrid Data Pipeline](#).

The following tables describe parameters available on the **General** and **OData** tabs of a JDBC Data Source dialog.

- **General tab**
Table 19: General tab connection parameters for JDBC

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Driver Class</td>
<td>The name of the class of the third party driver which is plugged into Hybrid Data Pipeline.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
User Id, Password | The login credentials used to connect to the JDBC database. A user name and password is required if user ID/password authentication is enabled on your database. Contact your system administrator to obtain your user name.

**Note:** By default, the password is encrypted.

By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye 🔄 icon. Click the icon again to conceal the password.

Connector ID | A general description of the data source.

Metadata Exposed Schemas | Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

```
<schema>
```

Where:

```
<schema>
```

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

Connection URL | The URL used by the third party driver to make a JDBC connection. This includes connection specific information like server name, the port to connect to, the database etc.

### OData tab

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 20: OData tab connection parameters for JDBC

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
Specifities the base URI for the OData feed to access your data source, for example, `https://hybridpipe.operations.com/api/odata/<DataSourceName>`. You can copy the URI and paste it into your application's OData configuration.

The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding `/$metadata` to the service root URI.

Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.

See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.

Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

**Valid Values:**

When set to 1, session caching is enabled. This provides better performance for production.

When set to 0, session caching is disabled. Use this value when you are configuring the data source.

**Default:** 1
### Field | Description
--- | ---
**Page Size** | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the `$top` and `$skip` parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.

**Valid Values:** 0 | \( n \)

where \( n \) is an integer from 1 to 10000.

When set to 0, the server default of 2000 is used.

**Default:** 0

**Refresh Result** | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.

**Valid Values:**

- When set to 0, the OData service caches the first page of results.
- When set to 1, the OData service re-executes the query.

**Default:** 1

**Inline Count Mode** | Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.

**Valid Values:**

- When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.
- When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

**Default:** 1
### Top Mode

Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**
- Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.
- Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.

**Default:** 0

### OData Read Only

Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.

Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.

**Valid Values:**
- **ON** | **OFF**

When **ON** is selected, OData access is restricted to read-only mode.

When **OFF** is selected, write operations can be performed on the OData service.

**Default:** OFF

---

### Google Analytics parameters

**Important:** Before you can proceed with creating a data source, an administrator must register Hybrid Data Pipeline as a client application with the Google Analytics API and create an OAuth application object using the Hybrid Data Pipeline OAuth applications API. For detailed information, see [Integrating Hybrid Data Pipeline with a Google OAuth 2.0 authorization flow to access Google Analytics](#).

The following tables describe parameters available on the tabs of a Google Analytics Data Source setup dialog:

- General tab
- OData tab
- Mapping tab
- Advanced tab
General tab

Create Google Analytics Data Source

- Data Source Name
- Description
- OAuth Profile Name
- Default View Name
- Segment
- Start Date
- End Date
Table 21: General tab connection parameters for Google Analytics

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>OAuth Profile Name</td>
<td>An OAuth profile contains access and refresh tokens generated by Google. These tokens allow Hybrid Data Pipeline to access the Google Analytics API on your behalf. You must either select an existing profile or create a new one.</td>
</tr>
<tr>
<td></td>
<td>For an existing OAuth profile, select the profile from the OAuth Profile Name drop-down list. The Default View Name and Segment fields will automatically populate.</td>
</tr>
<tr>
<td></td>
<td>To create a new profile, you must have administrative privileges on the Google Analytics project. To begin, click Create OAuth Profile Name, enter a profile name and click Create. A Google authorization pop-up window appears. In the authorization window, enter the required Google credentials and click Allow. Google Analytics supplies Hybrid Data Pipeline with access and refresh tokens. Then, you are returned to the General tab. Click Save to save these changes to the data source.</td>
</tr>
<tr>
<td></td>
<td>See the following topics for further details: Creating an OAuth profile, Renaming an OAuth profile, Deleting an OAuth profile, and Refreshing stale access and refresh tokens.</td>
</tr>
<tr>
<td>Default View Name</td>
<td>A view that belongs to your Google Analytics account. Select a view from the drop-down list.</td>
</tr>
<tr>
<td>Segment</td>
<td>A segment that belongs to your Google Analytics account. Select a segment from the drop-down list.</td>
</tr>
<tr>
<td>Start Date</td>
<td>The start date for fetching Google Analytics data (inclusive). You can enter a specific date in YYYY-MM-DD format, or select a date, using the calendar icon. Alternatively, select a relative value (Today, Yesterday, or N Days Ago, where N is a positive integer). The default is 30 days prior to the current date.</td>
</tr>
<tr>
<td>End Date</td>
<td>The end date for fetching Google Analytics data. You can enter a specific date in YYYY-MM-DD format, or select a date, using the calendar icon. Alternatively, select a relative value from the drop-down list (Today, Yesterday, or N Days Ago, where N is a positive integer). The end date must always be later than the start date, if a start date is specified.</td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see "Formulating queries" under Querying with OData.
Table 22: OData tab connection parameters for Google Analytics

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
**OData Access URI**

Specifies the base URI for the OData feed to access the data source, for example, https://example.com:8443/api/odata4/<datasourcename>. You can copy the URI and paste it into your application's OData configuration.

The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding $metadata to the service root URI.

**Schema Map**

Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.

See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.

**Data Source Caching**

Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

**Valid Values:**

When set to 1, session caching is enabled. This provides better performance for production.

When set to 0, session caching is disabled. Use this value when you are configuring the data source.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Page Size   | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** $0 | $n  
where $n$ is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0                                                                                           |
| Refresh Result | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1                                                                                           |
### Inline Count Mode

Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.

**Valid Values:**

When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.

When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

**Default:** 1

### Top Mode

Indicates how requests typically use `$top` and `$skip` for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**

Set to 0 when the application generally uses `$top` to limit the size of the result and rarely attempts to get additional entities by combining `$top` and `$skip`.

Set to 1 when the application uses `$top` as part of client-driven paging and generally combines `$top` and `$skip` to page through the result.

**Default:** 0

### Mapping tab

The Mapping tab enables you to create relational tables in Hybrid Data Pipeline and map them to Metrics and Dimensions in your Google Analytics data source.
Table 23: Mapping tab connection parameters for Google Analytics

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map Name</td>
<td>Optional name of the map definition that Hybrid Data Pipeline uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map. If you want to name the map yourself, enter a unique name.</td>
</tr>
</tbody>
</table>
### Refresh Schema

The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects.

**Valid Values:**

- **When set to ON,** the connectivity service attempts to refresh the schema.
- **When set to OFF,** the connectivity service does not attempt to refresh the schema.

**Default:**

OFF

**Notes**

- You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.
- Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.
- If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.

### Create Mapping

Determines whether the Google Analytics table mapping files are to be (re)created.

Hybrid Data Pipeline automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

**Table 24: Valid values for Create Map field**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the <strong>Map Name</strong> field, the name will be a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong>.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The Hybrid Data Pipeline connectivity service uses a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to <strong>Force New</strong> indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Add Tables</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>A set of tables to work with your Google Analytics account. To create configuration tables that use different combinations of Metrics and Dimensions, click <strong>Configure Logical Schema</strong>. In the <strong>Configure Logical Schema</strong> screen, click <strong>Create Table</strong> and enter a name for the table. <strong>Dimensions and Metrics</strong> screen, select the metrics that you want to add to the table. You can select metrics across multiple dimensions. Each metric gets added as a column in the table. Finally, click <strong>Save &amp; Close</strong>.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>For more information, see <a href="#">Adding Google Analytics tables</a> and <a href="#">Using Google Analytics</a>.</td>
<td></td>
</tr>
</tbody>
</table>
| **Show Deprecated Objects**  | Defines whether Hybrid Data Pipeline shows deprecated objects. Google Analytics marks dimensions and metrics as deprecated as an indication that they plan to remove support for those objects. By default, the Hybrid Data Pipeline connectivity service does not expose these deprecated objects. Set the value to **ON** while you work on rewriting your queries and table definitions to migrate from the deprecated objects. Once the queries and table definitions are fixed, change the setting for the map option back to **OFF**. **Valid Values:**  
  
  **ON | OFF**  
  
  If set to **ON**, Hybrid Data Pipeline includes deprecated objects in the relational model.  
  If set to **OFF**, Hybrid Data Pipeline does not include deprecated objects in the relational model. **Default:** **OFF** |
| **Show Internal Tables**     | Defines how Hybrid Data Pipeline shows internal tables. **Valid Values:**  
  
  **ON | OFF**  
  
  If set to **ON**, Hybrid Data Pipeline shows the "Data" table.  
  If set to **OFF**, Hybrid Data Pipeline does not show the "Data" table. **Default:** **OFF** |
| **Subtract Tables**          | Defines a comma-separated list of tables that should be hidden from the user's view. This feature is useful if you want to define your own tables instead of using some of the tables that are supplied with the data store, or to limit access to certain tables so that the user does not see them. For example, enter `adSense,adWords`.  
  
  `subtractTable` can be used both for the pseudo-tables in Google Analytics that are derived from the Data system table, and also for the regular management tables such as `Goal` or `Account`. |
Advanced tab

Table 25: Advanced tab connection parameters for Google Analytics

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Default Query Options        | A semi-colon delimited list of default values for the WHERE clauses within the connection. Specifying mandatory values such as `startDate`, `endDate`, and `viewId` in this parameter makes the queries simpler. For example, the query `SELECT * FROM Overview` returns only results from the specified period. **Valid Values:**
  
  `(key=value[;key=value])`
  
  **Where:**
  
  `key`
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>is one of the following values:</td>
</tr>
<tr>
<td></td>
<td>If set to <code>startDate</code>, specifies the starting date for the query (inclusive). The default is thirty days prior to the current date, expressed as <code>30daysAgo</code>.</td>
</tr>
<tr>
<td></td>
<td>If set to <code>endDate</code>, the ending date for the query (inclusive). This defaults to <code>yesterday</code>.</td>
</tr>
<tr>
<td></td>
<td>The syntax for <code>startDate</code> and <code>endDate</code> values is as follows:</td>
</tr>
<tr>
<td></td>
<td>• a date in <code>YYYY-MM-DD</code> format</td>
</tr>
<tr>
<td></td>
<td>• the word &quot;today&quot; for the current date</td>
</tr>
<tr>
<td></td>
<td>• the word &quot;yesterday&quot; for the prior date</td>
</tr>
<tr>
<td></td>
<td>• <code>#daysAgo</code>, where <code>#</code> is some positive integer</td>
</tr>
<tr>
<td></td>
<td>If the key is <code>viewId</code>, the value is a comma-separated list of view Ids. There is no default; in order for <code>SELECT * FROM</code> to work for either &quot;Data&quot; or any of the pseudo-tables, this must be set either explicitly in a WHERE clause or via the defaultQueryOptions connection string option.</td>
</tr>
<tr>
<td></td>
<td><strong>Default</strong>: If no value is specified (the default), the connectivity service uses <code>startDate=30daysAgo;endDate=yesterday</code>.</td>
</tr>
<tr>
<td><strong>Extended Options</strong></td>
<td>Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:</td>
</tr>
<tr>
<td></td>
<td><code>Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]</code></td>
</tr>
<tr>
<td></td>
<td>If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values</strong>: <code>string</code></td>
</tr>
<tr>
<td></td>
<td><strong>Default</strong>: empty string</td>
</tr>
<tr>
<td><strong>Initialization String</strong></td>
<td>A semicolon delimited set of commands to be executed on the cloud data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.</td>
</tr>
<tr>
<td></td>
<td>The default is an empty string.</td>
</tr>
<tr>
<td><strong>Max Pooled Statements</strong></td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.</td>
</tr>
<tr>
<td><strong>Web Service Call Limit</strong></td>
<td>The maximum number of Web service calls allowed for a single SQL statement or metadata query. When set to 0, there is no limit on the number of Web service calls on a single connection that can be made when executing a SQL statement.</td>
</tr>
</tbody>
</table>
### Field: Web Service Fetch Size

Specifies the number of rows of data the Hybrid Data Pipeline connectivity service attempts to fetch for each call.

**Valid Values:**

- 0  
- \(x\)

If set to 0, the Hybrid Data Pipeline connectivity service attempts to fetch up to a maximum of 10000 rows. This value typically provides the maximum throughput.

If set to \(x\), the Hybrid Data Pipeline connectivity service attempts to fetch up to a maximum of the specified number of rows. Setting the value lower than 10000 can reduce the response time for returning the initial data. Consider using a smaller value for interactive applications only.

**Default:** 0

### Field: Web Service Retry Count

The number of times to retry a timed-out *Select* request. The **Web Service Timeout** parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 3.

### Field: Web Service Timeout

The time, in seconds, to wait before retrying a timed-out *Select* request. Valid only if the value of **Web Service Retry Count** is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.

### Field: Metadata Exposed Schemas

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the **SQL Editor**, the **Configure Schema Editor**, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the **Schema API** will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the **Metadata Exposed Schemas** option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

\(<\text{schema}>\)

**Where:**

\(<\text{schema}>\)

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

---

**See the steps for:**
Creating an OAuth profile

Take the following steps to create an OAuth profile. This procedure requires administrative privileges on the Google Analytics project.

1. Click Create OAuth Profile Name.
2. Enter a profile name in the Create OAuth Profile dialog. Then, click Create.
   
   A Google authorization pop-up window appears.
3. In the Google authorization pop-up window, enter the required Google credentials.
4. Click Allow.
   
   Google Analytics supplies Hybrid Data Pipeline with access and refresh tokens. Then, you are returned to the General tab.
5. Click Save to save these changes to the data source.

Renaming an OAuth profile

The drop-down list for the OAuth Profile Name field contains the names of previously-created OAuth profiles. You can rename an existing or previously-created profile. This procedure requires a user with administrative privileges on the Google Analytics project.

1. Open the drop-down list in the OAuth Profile Name field, and click the Rename icon next to the profile that you want to rename.
   
   The profile appears in Edit mode.
2. Change the name of the profile and click the Add icon next to the profile name.
   
   The profile is renamed and you are returned to the General tab.
3. Click Save.

Deleting an OAuth profile

The drop-down list for the OAuth Profile Name field contains the names of previously-created OAuth profiles. You can delete an unused or outdated profile. This procedure requires a user with administrative privileges on the Google Analytics project.

1. Open the drop-down list in the OAuth Profile Name field, and click the Delete icon next to the profile that you want to delete.
   
   The OAuth profile is removed from the drop-down list in the OAuth Profile Name field.
2. Click Save.
Refreshing stale access and refresh tokens

Access and refresh tokens may expire or be revoked. To refresh these tokens in the Web UI, you must open a data source that uses the profile and click Authorize with Google. As with creating a profile for the first time, you are redirected to Google where you must log in to the Google account. When you click Accept, new access and refresh tokens will be supplied to Hybrid Data Pipeline. You are then returned to the Hybrid Data Pipeline Web UI.

Adding Google Analytics tables

To determine the content of the Add Tables field:

1. Click Configure Logical Schema.
2. Select the tables from a schema. If no primary key is defined in the table, set the primary key by selecting a column in the table.
3. Click Save & Close. The JSON of the configured schema appears in the Add Tables field.

To add Google Analytics tables:

1. Click Configure Logical Schema.
2. Click Create Table.
3. Type a name for the table, or select a table from the drop-down list.
4. Select from the options under the Dimensions and Metrics headings. If no primary key is defined in the table, set the primary key by selecting a column in the table.
5. Click Save & Close. The JSON of the configured schema appears in the Add Tables field.

Using Google Analytics

Google Analytics is a service that generates detailed statistics about a website's traffic and traffic sources. But Google Analytics is not just a database. It is a multi-dimensional hypercube containing all kinds of measurements about traffic to a website.

When you connect to Google Analytics using Hybrid Data Pipeline, you can reach into this repository and flatten it into relational data that can be used with any ODBC or JDBC application.

Imagine a very small store of data about your website. For each hit, the Analytics software logs the date, language of user, country of origin, new or returning user, and their time on the site (in seconds).
Google Analytics collected data for our little web site over four days. The data is broken down by date, language, country and user type. And for each visit, we recorded the time spent on the site.

You can look at the time on site as a *measurement* or *metric*, and all of the other columns as dimensions.

Google Analytics works like our example. It aggregates information from your website, but measures hundreds of things, and categorizes them by hundreds of dimensions.

The query interface that Google provides allows you to fetch these metrics and group them. Because of the massive amount of information they store, their interface limits you to fetching at most ten metrics at a time, and grouped by no more than seven *dimensions*.

### Creating a query

Suppose you want to know how much time new visitors spent on the site. Your dimension is user type and your metric is time. You would get back two rows:

<table>
<thead>
<tr>
<th>new</th>
<th>2648</th>
</tr>
</thead>
<tbody>
<tr>
<td>returning</td>
<td>821.6</td>
</tr>
</tbody>
</table>

How much data you get back depends on how you ask for it. If you ask for two dimensions, you get even more data, because you get one row per permutation. Requesting how much time users have spent on each day, broken down by country, returns more rows:

<table>
<thead>
<tr>
<th>2014-01-01 en US new</th>
<th>5.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2014-01-01 nl DK new</td>
<td>90.4</td>
</tr>
<tr>
<td>2014-01-01 es ES new</td>
<td>24</td>
</tr>
<tr>
<td>2014-01-01 ja JP new</td>
<td>4.2</td>
</tr>
<tr>
<td>2014-01-01 es MX new</td>
<td>345.3</td>
</tr>
<tr>
<td>2014-01-01 ja JP returning</td>
<td>655.9</td>
</tr>
<tr>
<td>2014-01-02 en US new</td>
<td>45.7</td>
</tr>
<tr>
<td>2014-01-02 en US new</td>
<td>345.9</td>
</tr>
<tr>
<td>2014-01-02 es ES new</td>
<td>57.7</td>
</tr>
<tr>
<td>2014-01-02 en US new</td>
<td>6.8</td>
</tr>
<tr>
<td>2014-01-03 es MX new</td>
<td>876.1</td>
</tr>
<tr>
<td>2014-01-03 ja JP returning</td>
<td>5.7</td>
</tr>
<tr>
<td>2014-01-03 en GB new</td>
<td>5.6</td>
</tr>
<tr>
<td>2014-01-03 en US new</td>
<td>617.9</td>
</tr>
<tr>
<td>2014-01-03 en US returning</td>
<td>56.1</td>
</tr>
<tr>
<td>2014-01-04 es MX new</td>
<td>45.1</td>
</tr>
<tr>
<td>2014-01-04 jp JP new</td>
<td>178</td>
</tr>
<tr>
<td>2014-01-04 en US returning</td>
<td>103.9</td>
</tr>
</tbody>
</table>
Google Analytics Dashboard

This section assumes you have access to a Google Analytics Dashboard.

Go to http://www.google.com/analytics/ and choose [Google Analytics] from the drop-down menu in the upper right corner.

An outline of your views into your web properties appears. Choose a view and you see the Audience Overview, a graph with other metrics showing Sessions, Users, Pageviews, Pages/Session, Average Session Duration, Bounce Rate and Percent of New Sessions. In the lower right is a breakdown of sessions by language.

The DataDirect Hybrid Data Pipeline connectivity service defines a table called Overview for your Google Analytics Data Store that provides similar information. After connecting to Google Analytics, you can use the following query to give you the same numbers as the Audience Overview.

```
SELECT * FROM Overview WHERE viewId = 'ga:12345678'
```

You can copy the viewId from the URI in your browser. The URI will end in something like this: /visitors-overview/a99999999w00000000p12345678/. Copy the digits after the final "p", and prefix them with a "ga:" as the viewId.

<table>
<thead>
<tr>
<th>VIEWID</th>
<th>SEGMENTID</th>
<th>STARTDATE</th>
<th>ENDDATE</th>
<th>_BROWSER</th>
<th>_OPERATINGSYSTEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ga:12345678</td>
<td>NULL</td>
<td>&quot;2014-01-01&quot;</td>
<td>&quot;2014-01-30&quot;</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>
A simpler way to get the result is to use the defaultView connection option in your connection string. The name of the view is generally displayed in view control on the General tab of the setup dialog. Include that name in the defaultView connection option, and the connectivity service will look up the viewId for you. For example, if your view were named "web.mycompany.com blog", you could use the following connection string:

```
Connection c = DriverManager.getConnection
("jdbc:datadirect:googleanalytics:configOptions=(defaultView=web.mycompany.com blog);clientid=XXX;clientsecret=YYY;refreshtoken=ZZZ");
```

Now your query is simpler:

```
SELECT * FROM Overview
```

The remaining examples assume that you made this change.

To make the difference between metrics and dimensions a little more clear, in the driver we prefix all dimensions with an underscore.

Note that only one row was returned, and all of the dimensions came out as NULL. We have a special rule that says if you ask for all dimensions, like we just did with the SELECT *, then we get no dimensions. These values would match exactly what we see in the Google Analytics Audience Overview.

If we ask for the same set by language:

```
SELECT _LANGUAGE,SESSIONS FROM Overview
```

we get exactly what was in the lower-right corner of that dashboard page.

### Overview table

The entire data store of Google Analytics is available in a hidden table called Data. The Overview table is actually a small view into the Data table that has selected metrics and dimensions that are useful together.

Other tables, which are also subsets of Data, come predefined. These tables are listed on the Google Analytics Pseudo Tables page.

By default, the actual underlying Data table is hidden. The Data table has over 100 metrics and dimensions, Google limits the number of metrics (to 10) and dimensions (to 7) for each query. Hiding the table makes it less likely that users will submit a query such as `SELECT TOP 10 * FROM DATA`, which could return results that are not very useful.

The Data table can be made visible by adding `showInternalTables=1` to the Map Options.

After doing that, the following query would work the same way as the `SELECT FROM Overview` query.

```
SELECT _LANGUAGE,SESSIONS FROM Data
```

### Adding your own tables

Usually, you don't need to expose the Data table, because new pseudo-tables can be added with the `addTables` configuration option. Suppose you wanted to define a table that let you query sessions only by language and country. This piece of JSON defines the new table:

```
{"MyTable":["sessions","_language","_country"]}
```

You can add it to your connection string using the reference controls on the dialogs.
This adds a new pseudo-table named MyTable, and it now has three columns, plus the "automatic" columns of viewId, segmentId, startDate and endDate. Now instead of doing what we did, we can do the functionality equivalent:

```sql
SELECT _LANGUAGE, SESSIONS FROM MyTable
```

Because of this, it is typically not necessary to expose the Data table. (Note that we could have defined this table as just based on sessions and language. But remember the earlier rule that said that if you request all dimensions, we behave as if you had selected none. This means that both `SELECT _LANGUAGE, SESSIONS` and `SELECT *` would have all referenced one dimension, and therefore, it would have not broken the data down by language. There is no harm in adding extra dimensions to your definition.)

**Defining the columns**

You can use the Metadata table to define the columns in your pseudo-table. The Metadata table has the list of all of the metrics and dimensions. Use only the metrics and dimensions that are marked with a "PUBLIC" status. The Hybrid Data Pipeline connectivity service ignores metrics and dimensions with a "DEPRECATED" status, unless show Deprecated Objects = ON is added to the config options.

Not all combinations of metrics and dimensions are valid. Refer to the table called Incompatible. If you see a row in that table that contains both columns, it means they can't be used in the same query.

**Support for custom variables, metrics, and dimensions**

Custom Variables are defined on the client, and are basically key=value pairs. There are 5 available (50 for premium). They are set in the webpage by calling methods defined in ga.js. They are used only for Google Analytics before the upgrade to Universal Analytics.

Custom Metrics and Dimensions are defined solely on the server, and the names are available as metadata. There are 20 (200 for premium) of each available. They replace the concept of custom variables when the web properties are upgraded to Universal Analytics.

If you need access to three of the new tables, AccountUserLink, WebpropertyUserLink, and/or ProfileUserLink, your refresh token may have to be regenerated to get the new permission.

**Greenplum parameters**

The following tables describe parameters available on the tabs of a Greenplum Data Source setup dialog:

- General tab
- OData tab
- Security tab
- Advanced tab
General tab

Create Greenplum Data Source

Data Source Name *
Description
User ID
Password
Server Name *
Port Number
5432
Database *
Connector ID

Required Fields

Cancel  TEST  Save
## Table 26: General tab connection parameters for Greenplum

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Password</td>
<td>Specifies a case-sensitive password that is used to connect to your Greenplum database. A password is required if user ID/Password authentication is enabled on your database. Contact your system administrator to obtain your password. Note: By default, the password is encrypted. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The port number of the Greenplum server to which you want to connect.</td>
</tr>
<tr>
<td>Server Name</td>
<td>Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, GreenplumServer or 122.23.15.12. Valid Values: string where: string is a valid IP address or server name. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two.</td>
</tr>
</tbody>
</table>
The login credentials for your Greenplum server.

The Hybrid Data Pipeline connectivity service uses this information to connect to the data store. The administrator of the data store must grant permission to a user with these credentials to access the data store and the target data.

**Note:** You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data Source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline connectivity service.

### Security tab

**Create Greenplum Data Source**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crypto Protocol Version</td>
<td>Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.</td>
</tr>
<tr>
<td>Encryption Method</td>
<td></td>
</tr>
<tr>
<td>Host Name In Certificate</td>
<td></td>
</tr>
<tr>
<td>Validate Server Certificate</td>
<td></td>
</tr>
</tbody>
</table>

**Table 27: Security tab connection parameters for Greenplum**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crypto Protocol Version</td>
<td>Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.</td>
</tr>
<tr>
<td>Valid Values:</td>
<td></td>
</tr>
</tbody>
</table>
### Field

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
</table>

cryptographic_protocol [[,] cryptographic_protocol]...

where:

cryptographic_protocol

is one of the following cryptographic protocols:

<table>
<thead>
<tr>
<th>TLSv1</th>
<th>TLSv1.1</th>
<th>TLSv1.2</th>
</tr>
</thead>
</table>

The client must send the highest version that it supports in the client hello.

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.

**Example**

Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

TLSv1.1,TLSv1.2

Default: TLSv1, TLSv1.1, TLSv1.2

### Encryption Method

Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server.

**Valid Values:**

noEncryption | SSL

If set to noEncryption, data is not encrypted or decrypted.

If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.

**Note:**

- Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You might want to set a login timeout using the Login Timeout property to avoid problems when connecting to a server that does not support SSL.

- When SSL is enabled, the following properties also apply:

  - Host Name In Certificate
  - ValidateServerCertificate
  - Crypto Protocol Version

The default value is noEncryption.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Host Name In Certificate | Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.  
**Valid Values:**  
host_name | #SERVERNAME#  
where host_name is a valid host name.  
If host_name is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the Hybrid Data Pipeline connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.  
If #SERVERNAME# is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the Hybrid Data Pipeline connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.  
**Default:** Empty string |
| Validate Server Certificate |                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

**Valid Values:**

| ON | OFF |

If set to **ON**, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate parameter is specified, the connectivity service also validates the certificate using a host name. The Host Name In Certificate parameter is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the connectivity service is connecting to is the server that was requested.

If set to **OFF**, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any Java system properties.

**Default:** **ON**

### OData tab

The following table describes the controls on the **OData** tab. For information on using the **Configure Schema** editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see "Formulating queries" under Querying with OData.
OData tab connection parameters for Greenplum

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
### Field: OData Access URI

Specifies the base URI for the OData feed to access your data source, for example, https://hybridpipe.operations.com/api/odata/DataSourceName. You can copy the URI and paste it into your application's OData configuration.

The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.

### Field: Schema Map

Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.

See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.

### Field: Data Source Caching

Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

**Valid Values:**

When set to 1, session caching is enabled. This provides better performance for production.

When set to 0, session caching is disabled. Use this value when you are configuring the data source.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Page Size     | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
   **Valid Values:** $0 | n$n is an integer from 1 to 10000.  
   When set to 0, the server default of 2000 is used.  
   **Default:** 0 |
| Refresh Result| Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
   **Valid Values:**  
   When set to 0, the OData service caches the first page of results.  
   When set to 1, the OData service re-executes the query.  
   **Default:** 1 |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
   The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
   **Valid Values:**  
   When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
   When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
   **Default:** 1 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Top Mode            | Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.  

**Valid Values:**  
Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.  
Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.  
**Default:** 0 |
| OData Read Only     | Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.  
Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.  
**Valid Values:**  
ON | OFF  
When ON is selected, OData access is restricted to read-only mode.  
When OFF is selected, write operations can be performed on the OData service.  
**Default:** OFF |
Advanced tab

Create Greenplum Data Source

Table 29: Advanced tab connection parameters for Greenplum

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alternate Servers</td>
<td>Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property.</td>
</tr>
</tbody>
</table>
### Catalog Options

Determines which type of metadata information is included in result sets when an application calls `DatabaseMetaData` methods.

**Valid Values:**

2 | 4

If set to 2, the Hybrid Data Pipeline connectivity service queries database catalogs for column information.

If set to 4, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate `getColumns()` calls using the `ResultSetMetaData` object instead of querying database catalogs for column information. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using `getColumns()`. The argument to `getColumns()` must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the connectivity service reverts to the default behavior for `getColumns()` calls.

**Default:** 2

### Extended Options

Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.

**Valid Values:** `string`

**Default:** empty string
### Initialization String

A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.

**Syntax:**

```
command[[; command]...]
```

**Where:**

- `command`

is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of `SFORCE`:

```
InitializationString=(REFRESH SCHEMA SFORCE)
```

The default is an empty string.

### Load Balancing

Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the `AlternateServers` property.

**Valid Values:** **ON | OFF**

If set to **ON**, the connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service then randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established.

If set to **OFF**, the connectivity service does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

**Default:** **OFF**

**Notes**

- The Alternate Servers parameter specifies one or multiple alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers property.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Login Timeout         | The amount of time, in seconds, to wait for a connection to be established before timing out the connection request.  
  **Valid Values:**  
  
  0 | \(x\)  
  
  where \(x\) is a positive integer that represents a number of seconds.  
  
  If set to 0, the connectivity service does not time out a connection request.  
  
  If set to \(x\), the connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.  
  
  **Default:** 30                                                                                                                                                                                                                                                                                                                                                      |
| Max Pooled Statements | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.  
  
  The default value is 0.                                                                                                                                                                                                                                                                                                                                                   |
| Query Timeout         | Sets the default query timeout (in seconds) for all statements created by a connection.  
  **Valid Values:**  
  
  -1 | 0 | \(x\)  
  
  If set to –1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the Statement.setQueryTimeout() method.  
  
  If set to 0, the default query timeout is infinite (the query does not time out).  
  
  If set to \(x\), the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection option, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.  
  
  **Default:** 0                                                                                                                                                                                                                                                                                                                                                      |
Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.

**Valid Values:**

0 | 1

If set to 0 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getTableName() method may return an empty string for each column in the result set.

If set to 1 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the ResultSetMetaData.getSchemaName() and ResultSetMetaData.getCatalogName() methods are called if the connectivity service can determine that information.

**Default:** 0
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata Exposed Schemas</td>
<td>Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.</td>
</tr>
<tr>
<td></td>
<td><strong>Warning:</strong> This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.</td>
</tr>
<tr>
<td>Valid Values</td>
<td>&lt;schema&gt;</td>
</tr>
<tr>
<td></td>
<td>Where: &lt;schema&gt;</td>
</tr>
<tr>
<td></td>
<td>is the name of a valid schema on the backend data store.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> No schema is specified. Therefore, all schemas are exposed.</td>
</tr>
<tr>
<td>Transaction Error Behavior</td>
<td>Determines how the driver handles errors that occur within a transaction. When an error occurs in a transaction, the Greenplum server does not allow any operations on the connection except for rolling back the transaction.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>none</td>
</tr>
<tr>
<td></td>
<td>If set to none, the connectivity service does not roll back the transaction when an error occurs. The application must handle the error and roll back the transaction. Any operation on the statement other than a rollback results in an error.</td>
</tr>
<tr>
<td></td>
<td>If set to RollbackTransaction, the connectivity service rolls back the transaction when an error occurs. In addition to the original error message, the connectivity service posts an error message indicating that the transaction has been rolled back.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> RollbackTransaction</td>
</tr>
</tbody>
</table>

**Informix parameters**

The following tables describe parameters available on the tabs of an Informix Data Source dialog:

- General tab
- OData tab
• **Advanced tab**

**General tab**

Create Informix Data Source

<table>
<thead>
<tr>
<th><strong>Data Source Name</strong></th>
<th>Required Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>User ID</td>
<td></td>
</tr>
<tr>
<td>Password</td>
<td></td>
</tr>
<tr>
<td>Host Name</td>
<td></td>
</tr>
<tr>
<td>Port Number</td>
<td></td>
</tr>
<tr>
<td>Informix Server</td>
<td></td>
</tr>
<tr>
<td>Database</td>
<td></td>
</tr>
<tr>
<td>Connector ID</td>
<td></td>
</tr>
</tbody>
</table>

[Image of the Create Informix Data Source form]
## Table 30: General tab connection parameters for Informix

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premises Connector that is used to access the on-premise data source. Click the arrow and select the Connector that you want to use. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premises Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the drop-down list were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, InformixServer or 122.23.15.12. Valid Values: string where: string is a valid IP address or server name. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two.</td>
</tr>
<tr>
<td>Informix Server</td>
<td>The name of the Informix database server to which you want to connect.</td>
</tr>
<tr>
<td>Password</td>
<td>Specifies a case-sensitive password that is used to connect to your Informix database. A password is required if user ID/Password authentication is enabled on your database. Contact your system administrator to obtain your password. <strong>Note:</strong> By default, the password is encrypted. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
</tbody>
</table>
The port number of the Informix server to which you want to connect.

User Id

The login credentials for your Informix server.

The Hybrid Data Pipeline connectivity service uses this information to connect to the data store. The administrator of the data store must grant permission to a user with these credentials to access the data store and the target data.

**Note:** You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data Source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline connectivity service.

---

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 31: OData tab connection parameters for Informix

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Page Size    | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** $0 | $n  
where $n is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0 |
| Refresh Result | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1 |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
**Valid Values:**  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
**Default:** 1 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0                                                                ections 0 and $skip.</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>When <strong>ON</strong> is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When <strong>OFF</strong> is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
</tbody>
</table>
**Advanced tab**

*Create Informix Data Source*

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Alternate Servers | Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property.  
**Valid Values:**  
\[(servername1[:port1] [,servername2[:port2]]...)\]  
The server name (servername1, servername2, and so on) is required for each alternate server entry. Port number (port1, port2, and so on) is optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number is used. |
### Catalog Options

Determines which type of metadata information is included in result sets when an application calls DatabaseMetaDatam methods. To include multiple types of metadata information, add the sum of the values that you want to include. In this case, specify 6 to query database catalogs for column information and to emulate getColumns() calls.

**Valid Values:**

2 | 4

If set to 2, the connectivity service queries database catalogs for column information.
If set to 4, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate getColumns() calls using the ResultSetMetaData object instead of querying database catalogs for column information. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumns(). The argument to getColumns() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the Hybrid Data Pipeline connectivity service reverts to the default behavior for getColumns() calls.

**Default:** 2

### Code Page Override

The code page to be used by the Hybrid Data Pipeline connectivity service to convert Character and Clob data. The specified code page overrides the default database code page or column collation. All Character and Clob data that is returned from or written to the database is converted using the specified code page.

By default, the Hybrid Data Pipeline connectivity service automatically determines which code page to use to convert Character data. Use this parameter only if you need to change the connectivity service’s default behavior.

**Valid Values:**

```
string
```

where `string` is the name of a valid code page that is supported by your JVM. **CP950**.

**Default:** empty string

### Extended Options

Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.

**Valid Values:** `string`

**Default:** empty string
## Initialization String

A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.

**Syntax:**

```
command[, command]...
```

**Where:**

- `command`

  is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of **SFORCE**:

  ```
  InitializationString=(REFRESH SCHEMA SFORCE)
  ```

  The default is an empty string.

## Load Balancing

Determine whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property.

**Valid Values:**

- **ON**
- **OFF**

If set to **ON**, the connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established.

If set to **OFF**, the connectivity service does not use client load balancing and connects to each server based on their sequential order (primary server first, then alternate servers in the order they are specified).

**Default:** **OFF**

**Notes**

- The Alternate Servers connection parameter specifies one or multiple alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers parameter.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Login Timeout       | The amount of time, in seconds, that the Hybrid Data Pipeline connectivity service waits for a connection to be established before timing out the connection request.  
  **Valid Values:**  
  $0 \mid x$  
  where $x$ is a positive integer that represents a number of seconds.  
  If set to 0, the Hybrid Data Pipeline connectivity service does not time out a connection request.  
  If set to $x$, the connectivity service waits for the specified number of seconds before returning control to the application and returning a timeout error.  
  **Default:** 30 |
| Max Pooled Statements | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.  
  The default value is 0. |
| Query Timeout       | Sets the default query timeout (in seconds) for all statements created by a connection.  
  **Valid Values:**  
  $-1 \mid 0 \mid x$  
  If set to $-1$, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the connectivity service silently ignores calls to the :: Statement.setQueryTimeout() method.  
  If set to 0, Statement.setQueryTimeout() method., the default query timeout is infinite (the query does not time out).  
  If set to $x$, the Hybrid Data Pipeline connectivity service uses the value as Hybrid Data Pipeline the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection option, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.  
  **Default:** 0 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResultSet Meta Data Options</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.</td>
</tr>
<tr>
<td><strong>Valid Values</strong></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>If set to 0 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getTableName() method may return an empty string for each column in the result set.</td>
</tr>
<tr>
<td></td>
<td>If set to 1 and the ResultSetMetaData.getTableName() method is called, the connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the ResultSetMetaData.getSchemaName() and ResultSetMetaData.getCatalogName() methods are called if the Hybrid Data Pipeline connectivity service can determine that information.</td>
</tr>
<tr>
<td><strong>Default</strong>: 0</td>
<td></td>
</tr>
<tr>
<td>Metadata Exposed Schemas</td>
<td>Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.</td>
</tr>
<tr>
<td><strong>Warning</strong>: This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.</td>
<td></td>
</tr>
<tr>
<td><strong>Valid Values</strong></td>
<td>&lt;schema&gt;</td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td>&lt;schema&gt;</td>
</tr>
<tr>
<td></td>
<td>is the name of a valid schema on the backend data store.</td>
</tr>
<tr>
<td><strong>Default</strong>: No schema is specified. Therefore, all schemas are exposed.</td>
<td></td>
</tr>
</tbody>
</table>

**See the steps for:**

*How to create a data source in the Web UI* on page 225

**Microsoft Dynamics CRM parameters**

The following tables describe parameters available on the tabs of an on-premise **Data Source** dialog for Microsoft Dynamics’ CRM:
General tab

**Create Dynamics CRM Data Source**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
</tbody>
</table>
A general description of the data source.

A URL that can be used to connect to your organization’s SOAP service. To obtain this URL, sign into your organization’s CRM site using the browser. Select Settings. When you have selected the settings, select Customization. Then, select Developer Resources. An example of an Organization Service URL is https://mycompany.api.crm.dynamics.com/XRMServices/2011/Organization.svc

### Security tab

Create Dynamics CRM Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Method</td>
<td>Determines which authentication method the Hybrid Data Pipeline connectivity service uses when it establishes a connection.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td></td>
</tr>
<tr>
<td>Kerberos</td>
<td>At this time, the Hybrid Data Pipeline connectivity service always uses Kerberos authentication when it establishes a connection.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Domain Name | Specifies the domain of the network that Microsoft Dynamics CRM locates.
Service Principle Name | Specifies the service principal name to be used by the Hybrid Data Pipeline connectivity service for Kerberos authentication.

**Valid Values:**

- `string`

Where `string` is a valid service principal name. This name is case-sensitive.

#### OData tab

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
### Table 35: OData tab connection parameters for Microsoft Dynamics CRM

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OData Version</strong></td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td><strong>OData Access URI</strong></td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <code>https://hybridpipe.operations.com/api/odata/&lt;DataSourceName&gt;</code>. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding <code>/$metadata</code> to the service root URI.</td>
</tr>
<tr>
<td><strong>Schema Map</strong></td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See <a href="#">Configuring data sources for OData connectivity and working with data source groups</a> on page 622 for more information.</td>
</tr>
<tr>
<td><strong>Data Source Caching</strong></td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **Page Size**  | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side pagination works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** $0 | $n  
where $n is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0 |
| **Refresh Result** | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1 |
| **Inline Count Mode** | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
**Valid Values:**  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
**Default:** 1 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Top Mode           | Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.  
**Valid Values:**  
Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.  
Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.  
**Default:** 0 |
| OData Read Only    | Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.  
Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.  
**Valid Values:**  
ON | OFF  
When ON is selected, OData access is restricted to read-only mode.  
When OFF is selected, write operations can be performed on the OData service.  
**Default:** OFF |

**Mapping tab**

Create Dynamics CRM Data Source

<table>
<thead>
<tr>
<th>General</th>
<th>Security</th>
<th>OData</th>
<th>Mapping</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

The default values for advanced mapping fields are appropriate in many cases. However, if your organization wants to strip custom prefixes or enable uppercase identifiers, you might want to change map option settings. Understanding how Hybrid Data Pipeline creates and uses maps will help you choose the appropriate values.

The first time you save and test a connection, a map for that data store is created. Once a map is created, you cannot change the map options for that Data Source definition unless you also create a new map. For example, suppose a map is created with Strip Custom Prefix set to new, test. Later, you change the value to new, abc. You will get an error saying the configuration options do not match. Simply change the value of the Create Map option to force creation of a new map.

The following table describes the mapping options that apply to Microsoft Dynamics CRM. Click the + next to Set Map Options to display the optional fields.

**Note:** Map creation is an expensive operation. In most cases, you will only want to re-create a map if you need to change mapping options.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Mapping</td>
<td>Determines whether the Microsoft Dynamics CRM table mapping files are to be (re)created. The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.</td>
</tr>
</tbody>
</table>

**Table 37: Valid values for Create Map field**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the Map Name field, the name will be a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong>.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to Force New indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map Name</td>
<td>Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map. If you want to name the map yourself, enter a unique name.</td>
</tr>
</tbody>
</table>
The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects.

Valid Values:
When set to ON, the connectivity service attempts to refresh the schema.
When set to OFF, the connectivity service does not attempt to refresh the schema.

Default
OFF

Notes
• You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.
• Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.
• If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.
Microsoft Dynamics CRM data stores treat the creation of standard and custom objects differently. Objects you create in your organization are called custom objects, and the objects already created for you by the data store administrator are called standard objects.

When you create custom objects such as tables and columns, Microsoft Dynamics CRM prepends a string of lowercase characters, immediately followed by an underscore to the name of the custom object, for example, \texttt{new\_}. You can change this custom prefix, and define one or multiple prefixes for the same Microsoft Dynamics CRM instance. This custom prefix can be stripped from the table names, allowing you to make queries without adding the prefix.

For example, a Microsoft Dynamics CRM user who creates a custom object named \texttt{emp} might expect to be able to query the table using that name. However, because Microsoft Dynamics CRM has added the \texttt{new\_} prefix, the query must include it in the object name, for example, \texttt{SELECT * FROM new\_emp}. By default, the map strips the prefix, so in this example, the user can make the query without adding the prefix (\texttt{SELECT * FROM emp}).

**Valid Values:**

- If set to \texttt{new} (the default), the prefix \texttt{new\_} is stripped.
- If a comma-separated string, for example, \texttt{new,test,abc} is specified, the specified prefixes are stripped.
- If the special value \texttt{<none>} is specified, no prefixes are stripped.

The angle brackets are required for this special value. If you are disabling the option via an XML-based configuration, you must explicitly add the value as \texttt{&lt;none&gt;};

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Strip Custom Prefix    | Microsoft Dynamics CRM data stores treat the creation of standard and custom objects differently. Objects you create in your organization are called custom objects, and the objects already created for you by the data store administrator are called standard objects. When you create custom objects such as tables and columns, Microsoft Dynamics CRM prepends a string of lowercase characters, immediately followed by an underscore to the name of the custom object, for example, \texttt{new\_}. You can change this custom prefix, and define one or multiple prefixes for the same Microsoft Dynamics CRM instance. This custom prefix can be stripped from the table names, allowing you to make queries without adding the prefix. For example, a Microsoft Dynamics CRM user who creates a custom object named \texttt{emp} might expect to be able to query the table using that name. However, because Microsoft Dynamics CRM has added the \texttt{new\_} prefix, the query must include it in the object name, for example, \texttt{SELECT * FROM new\_emp}. By default, the map strips the prefix, so in this example, the user can make the query without adding the prefix (\texttt{SELECT * FROM emp}). Valid Values:  
  - If set to \texttt{new} (the default), the prefix \texttt{new\_} is stripped.  
  - If a comma-separated string, for example, \texttt{new,test,abc} is specified, the specified prefixes are stripped.  
  - If the special value \texttt{<none>} is specified, no prefixes are stripped. The angle brackets are required for this special value. If you are disabling the option via an XML-based configuration, you must explicitly add the value as \texttt{&lt;none&gt;}; |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Defines how Hybrid Data Pipeline maps identifiers. By default, all unquoted identifier names are mapped to uppercase. Identifiers are object names. Classes, methods, variables, interfaces, and database objects, such as tables, views, columns, indexes, triggers, procedures, constraints, and rules, can have identifiers.</td>
</tr>
</tbody>
</table>

**Valid Values:**

When set to **ON**, the connectivity service maps all identifier names to uppercase.

When set to **OFF**, Hybrid Data Pipeline maps identifiers to the mixed case name of the object being mapped. If mixed case identifiers are used, those identifiers must be quoted in SQL statements, and the case of the identifier, must exactly match the case of the identifier name.

**Note:** When object names are passed as arguments to catalog functions, the case of the value must match the case of the name in the database. If an unquoted identifier name was used when the object was created, the value passed to the catalog function must be uppercase because unquoted identifiers are converted to uppercase before being used. If a quoted identifier name was used when the object was created, the value passed to the catalog function must match the case of the name as it was defined. Object names in results returned from catalog functions are returned in the case that they are stored in the database.

For example, if **Uppercase Identifiers** is set to **ON**, to query the **Account** table you would need to specify:

```sql
SELECT "id", "name" FROM "Account"
```

**Default:** **ON**
**Advanced tab**

![Advanced tab screenshot](image)

**Table 38: Advanced tab connection parameters for Microsoft Dynamics CRM**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Extended Options | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  

```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  

**Valid Values:** string  

**Default:** empty string
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Initialization String     | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed. Syntax:  

command[; command]...

Where:

command

is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:

InitializationString=(REFRESH SCHEMA SFORCE)

The default is an empty string. |
| Login Timeout             | The amount of time, in seconds, to wait for a connection to be established before timing out the connection request. If set to 0, the Hybrid Data Pipeline connectivity service does not time out a connection request.                                                                                                                                                                                                                                  |
| Max Pooled Statements     | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application. The default value is 0.                                                                                                                                                                                                                           |
| Read Only                 | Sets the connection to read-only mode, indicating that the data store can be read but not updated. By default, this option is set to OFF.                                                                                                                                                                                                                                                                                                                                 |
| Web Service Call Limit    | The maximum number of Web service calls allowed to the data store for a single SQL statement or metadata query. The default value of 0 implies there is no limit.                                                                                                                                                                                                                                                                                                           |
### Web Service Batch Size

The maximum number of requests to be batched together in a single Web service call. If configured for 0, the connectivity service uses the default value 1000. Valid values are from 0 to 1000.

### Metadata Exposed Schemas

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

#### Valid Values

- `<schema>`

Where:

- `<schema>`

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

---

**See the steps for:**

*How to create a data source in the Web UI on page 225*

---

**Microsoft SQL Server parameters**

The following tables describe parameters available on the tabs of a Microsoft SQL Server **Data Source** dialog:

- **General tab**
- **OData tab**
- **Security tab**
- **Data Types tab**
- **Advanced tab**

The connection parameters also apply to Microsoft Azure SQL Database, unless specifically noted.
General tab

Create SQL Server Data Source

- **Data Source Name**
- **Description**
- **User ID**
- **Password**
- **Server Name**
- **Port Number**
- **Database**
- **Connector ID**

Required Fields
**Table 39: General tab connection parameters for Microsoft SQL Server**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server. If not specified, the default database for your login is used.</td>
</tr>
<tr>
<td>Password</td>
<td>A case-sensitive password that is used to connect to your Microsoft SQL Server database or Azure instance. A password is required if user ID/password authentication is enabled on your database. Contact your system administrator to obtain your password. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password. <strong>Note:</strong> By default, the password is encrypted.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The TCP port of the primary database server that is listening for connections to the database or Azure instance.</td>
</tr>
<tr>
<td>Server Name</td>
<td>The name of the server on which the SQL Server database to connect to is running. This is the fully qualified host name by which the server is accessed via the WAN. For example, mysqlserver.integration.mycorp.com. To connect to an Always On Availability group, the virtual network name (VNN) of the availability group listener must be specified.</td>
</tr>
<tr>
<td>User Id</td>
<td>The login credentials for your Microsoft SQL Server data store account. The Hybrid Data Pipeline connectivity service uses this information to connect to the data store. The administrator of the data store must grant permission to a user with these credentials to access the data store and the target data. <strong>Note:</strong> You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data Source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline connectivity service.</td>
</tr>
</tbody>
</table>
The unique identifier of the On-Premises Connector that is used to access the on-premise data source. Click the arrow and select the Connector that you want to use. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector.

If you have not installed an On-Premises Connector, and no Connectors have been shared with you, this field and drop-down list are empty.

If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the drop-down list were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).

### Security tab

The following table describes the controls on the Security tab.

#### Create SQL Server Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premises Connector that is used to access the on-premise data source. Click the arrow and select the Connector that you want to use. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premises Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the drop-down list were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
</tbody>
</table>

![Security tab](image)
Table 40: Security tab connection parameters for Microsoft SQL Server

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Method</td>
<td>Determines which authentication method the connectivity service uses when establishing a connection. If the specified authentication method is not supported by the database server, the connection fails and the connectivity service returns an error.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>ntlmjava</td>
</tr>
<tr>
<td></td>
<td>If set to ntlmjava, the connectivity service uses NTLM authentication, but requires a user ID and password to be specified. You must specify the name of the domain server that administers the database. You can specify the domain server using the Domain property. If the Domain property is not specified, the connectivity service tries to determine the domain server from the User property. If the connectivity service cannot determine the domain server name, it returns an error.</td>
</tr>
<tr>
<td></td>
<td>If set to ntlm2java, the connectivity service uses NTLMv2 authentication, but requires a user ID and password to be specified. You must specify the name of the domain server that administers the database. You can specify the domain server using the Domain property. If the Domain property is not specified, the connectivity service tries to determine the domain server from the User property. If the connectivity service cannot determine the domain server name, it returns an error. This value is supported for Windows and UNIX/Linux clients.</td>
</tr>
<tr>
<td></td>
<td>If set to userIdPassword, the connectivity service uses SQL Server authentication when establishing a connection. If a user ID is not specified, the connectivity service returns an error.</td>
</tr>
<tr>
<td></td>
<td>If set to ActiveDirectoryPassword, the connectivity service uses an Active Directory principal name and password to connect to the SQL Database or Azure instance. If a user ID is not specified, the connectivity service returns an error.</td>
</tr>
<tr>
<td></td>
<td>The default value is userIdPassword.</td>
</tr>
<tr>
<td>Crypto Protocol Version</td>
<td>Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>cryptographic_protocol [ [, cryptographic_protocol ] ... ]</td>
</tr>
<tr>
<td></td>
<td>where:</td>
</tr>
<tr>
<td></td>
<td>cryptographic_protocol</td>
</tr>
<tr>
<td></td>
<td>is one of the following cryptographic protocols:</td>
</tr>
<tr>
<td></td>
<td>TLSv1</td>
</tr>
<tr>
<td></td>
<td>The client must send the highest version that it supports in the client hello.</td>
</tr>
<tr>
<td><strong>Note:</strong> Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.</td>
</tr>
<tr>
<td></td>
<td>TLSv1.1, TLSv1.2</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> TLSv1, TLSv1.1, TLSv1.2</td>
</tr>
<tr>
<td><strong>Domain</strong></td>
<td>Specifies the name of the domain server that administers the database. Set this parameter only if you are using NTLM authentication (Authentication Method=ntlmjava). If the Domain property is unspecified, the connectivity service tries to determine the domain server name from the User property.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>where string is the name of the domain server.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> empty string</td>
</tr>
<tr>
<td><strong>Encryption Method</strong></td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>noEncryption</td>
</tr>
<tr>
<td></td>
<td>If set to noEncryption, data is not encrypted or decrypted.</td>
</tr>
<tr>
<td></td>
<td>If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>- Connection hangs can occur when the connectivity service is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the Login Timeout parameter to avoid problems when connecting to a server that does not support SSL.</td>
</tr>
<tr>
<td></td>
<td>- When SSL is enabled, the following properties also apply:</td>
</tr>
<tr>
<td></td>
<td>- Host Name In Certificate</td>
</tr>
<tr>
<td></td>
<td>- ValidateServerCertificate</td>
</tr>
<tr>
<td></td>
<td>- Crypto Protocol Version</td>
</tr>
<tr>
<td></td>
<td><strong>The default value is noEncryption.</strong></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HostNameInCertificate</td>
<td>Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>host_name</td>
</tr>
<tr>
<td></td>
<td>where host_name is a valid host name.</td>
</tr>
<tr>
<td></td>
<td>If host_name is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.</td>
</tr>
<tr>
<td></td>
<td>If #SERVERNAME# is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the Hybrid Data Pipeline connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.</td>
</tr>
<tr>
<td></td>
<td>The default is an empty string.</td>
</tr>
<tr>
<td>ValidateServerCertificate</td>
<td></td>
</tr>
</tbody>
</table>
Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

**Valid values:**

| ON | OFF |

If set to **ON**, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate parameter is specified, the connectivity service also validates the certificate using a host name. The Host Name In Certificate parameter is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the connectivity service is connecting to is the server that was requested.

If set to **OFF**, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any Java system properties.

**Default:** ON

### OData tab
The following table describes the controls on the **OData** tab. For information on using the **Configure Schema** editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 41: OData tab connection parameters for Microsoft SQL Server

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
</tbody>
</table>
| Page Size                 | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages. **Valid Values:** 0 | n  
  where n is an integer from 1 to 10000.  
  When set to 0, the server default of 2000 is used. **Default:** 0                                                                                                                                                                                                 |

Creating data sources with the Web UI
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh Result</td>
<td>Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>When set to 0, the OData service caches the first page of results.</td>
</tr>
<tr>
<td></td>
<td>When set to 1, the OData service re-executes the query.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Inline Count Mode</td>
<td>Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.</td>
</tr>
<tr>
<td></td>
<td>The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(<em>) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(</em>) aggregate; however, it may have a longer initial response time for the first page if the result is large.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.</td>
</tr>
<tr>
<td></td>
<td>When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td><strong>ON</strong></td>
</tr>
<tr>
<td></td>
<td>When <strong>ON</strong> is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When <strong>OFF</strong> is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> <strong>OFF</strong></td>
</tr>
</tbody>
</table>
Chapter 3: Using Hybrid Data Pipeline

DataTypes tab

Create SQL Server Data Source

- Data Time Input Parameter Type: auto
- Describe Input Parameters: noDescribe
- FetchTWFSasTTime: off
- FetchTSWTZ as Timestamp: off
- String Input Parameter Type: nvarchar
- Truncate Fractional Seconds: on
- XML Describe Type

[Buttons: Cancel, TEST, Save]
Table 42: DataTypes tab connection parameters for Microsoft SQL Server

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Date/Time/Timestamp Input Parameters | Specifies how the Hybrid Data Pipeline connectivity service describes the data type for Date/Time/Timestamp input parameters. This parameter only applies to connections to Microsoft SQL Server 2008 and higher and Microsoft Azure SQL Database. For connections to prior versions of Microsoft SQL Server, the Hybrid Data Pipeline connectivity service always describes Date/Time/Timestamp input parameters as datetime.  

**Valid values:**  
  - auto  
  - dateTime  
  - dateTimeOffset  

If set to auto, the Hybrid Data Pipeline connectivity service uses the following rules to describe the data type of Date/Time/Timestamp input parameters:  
  - If an input parameter is set using setDate(), the Hybrid Data Pipeline connectivity service describes it as date.  
  - If an input parameter is set using setTime(), the Hybrid Data Pipeline connectivity service describes it as time.  
  - If an input parameter is set using setTimestamp(), the Hybrid Data Pipeline connectivity service describes it as dateTimeOffset.  

If set to dateTime, the Hybrid Data Pipeline connectivity service describes Date/Time/Timestamp input parameters as dateTime.  

If set to dateTimeOffset, the Hybrid Data Pipeline connectivity service describes Date/Time/Timestamp input parameters as dateTimeOffset.  

**Default:** auto
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Describe Input Parameters</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service attempts to determine, at execute time, which data type to use to send input parameters to the database server. Sending parameters as the data type the database expects improves performance and prevents locking issues caused by data type mismatches. <strong>Valid values:</strong> noDescribe</td>
</tr>
<tr>
<td></td>
<td>If set to noDescribe, the Hybrid Data Pipeline connectivity service sends String and Date/Time/Timestamp input parameters to the server as specified by the String Input Parameter Type and DateTime Input Parameter Type parameters. If set to describeIfString, the Hybrid Data Pipeline connectivity service submits a request to the database to describe String input parameters. The Hybrid Data Pipeline connectivity service uses the data types that it returns to determine whether to describe the String input parameters as nvarchar or varchar. If this operation fails, the connectivity service sends String input parameters to the server as specified by the String Input Parameter Type parameter. If set to describeIfDateTime, the Hybrid Data Pipeline connectivity service submits a request to the database to describe Date/Time/Timestamp input parameters. The connectivity service uses the data types that it returns to determine how to describe the Date/Time/Timestamp input parameters. If this operation fails, the connectivity service sends Date/Time/Timestamp input parameters to the server as specified by the DateTime Input Parameter Type connection parameter. If set to describeAll, the Hybrid Data Pipeline connectivity service submits a request to the database to describe both String and Date/Time/Timestamp input parameters and uses the data types that it returns to determine which data type to use to describe the input parameters. If this operation fails, the connectivity service sends String input parameters to the server as specified by the String Input Parameter Type parameter and sends Date/Time/Timestamp input parameters to the server as specified by the Date Time Input Parameter connection parameter. <strong>Default:</strong> noDescribe</td>
</tr>
<tr>
<td>Fetch TWFS AsTime</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service returns column values with the time data type as the JDBC data type TIME or TIMESTAMP. Supported only for Microsoft SQL Server 2008 and higher. <strong>Valid values:</strong> ON</td>
</tr>
<tr>
<td></td>
<td>If set to ON, the Hybrid Data Pipeline connectivity service returns column values with the time data type as the JDBC data type TIME. The fractional seconds portion of the value is truncated. If set to OFF, the Hybrid Data Pipeline connectivity service returns column values with the time data type as the JDBC data type TIMESTAMP. The fractional seconds portion of the value is preserved. Time columns are not searchable when they are described and fetched as timestamp. <strong>Default:</strong> OFF</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| FetchTSWTZAs Timestamp | Determines whether column values with the datetimeoffset data type are returned as a JDBC VARCHAR or TIMESTAMP data type. <br>This parameter only applies to connections to Microsoft SQL Server 2008 and higher and Microsoft Azure SQL Database.  
**Valid values:** ON | OFF  
If set to ON, column values with the datetimeoffset data type are returned as a JDBC TIMESTAMP data type.  
If set to OFF, column values with the datetimeoffset data type are returned as a JDBC VARCHAR data type.  
**Default:** OFF |
| String Input Parameter Type | Determines whether the Hybrid Data Pipeline connectivity service sends String input parameters to the database in Unicode or in the default character encoding of the database.  
**Valid values:** nvarchar | varchar  
If set to nvarchar, the Hybrid Data Pipeline connectivity service sends String input parameters to the database in Unicode.  
If set to varchar, the Hybrid Data Pipeline connectivity service sends String input parameters to the database in the default character encoding of the database. This value can improve performance because the server does not need to convert Unicode characters to the default encoding.  
**Notes**  
- When set to nvarchar and a value is specified for the CodePageOverride parameter, this parameter is ignored and a warning is generated.  
**Default:** nvarchar |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Truncate Fractional Seconds  | Determines whether the Hybrid Data Pipeline connectivity service truncates timestamp values to three fractional seconds. For example, a value of the datetime2 data type can have a maximum of seven fractional seconds.  
**Valid values:**  
ON | OFF  
If set to ON, the Hybrid Data Pipeline connectivity service truncates all timestamp values to three fractional seconds.  
If set to OFF, the Hybrid Data Pipeline connectivity service does not truncate fractional seconds.  
**Default:** ON |

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| XML Describe Type          | Determines whether the Hybrid Data Pipeline connectivity service maps XML data to the LONGVARCHAR or LONGVARBINARY data type.  
**Valid values:**  
longvarchar | longvarbinary  
If set to longvarchar, the Hybrid Data Pipeline connectivity service maps XML data to the LONGVARCHAR data type.  
If set to longvarbinary, the Hybrid Data Pipeline connectivity service maps XML data to the LONGVARBINARY data type.  
**Default:** empty string |
**Advanced tab**

Table 43: Advanced tab connection parameters for Microsoft SQL Server

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Alternate Servers    | Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property.  
**Valid Values:**  
\( \text{servername1[:port1]}, \text{servername2[:port2]} \ldots \) |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Always Report Trigger Results</td>
<td>The server name (<em>servername1</em>, <em>servername2</em>, and so on) is required for each alternate server entry. Port number (<em>port1</em>, <em>port2</em>, and so on) is optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number is used. <strong>Default:</strong> None</td>
</tr>
</tbody>
</table>
| Application Intent | Determines how the Hybrid Data Pipeline connectivity service reports results that are generated by database triggers (procedures that are stored in the database and executed, or fired, when a table is modified). For Microsoft SQL Server 2005 and higher and Azure, this includes triggers that are fired by Data Definition Language (DDL) events. **Valid values:** **ON | OFF**  
If set to **ON**, the Hybrid Data Pipeline connectivity service returns all results, including results that are generated by triggers. Multiple trigger results are returned one at a time. You can use the SQLMoreResults function to return individual trigger results. Warnings and errors are reported in the results as they are encountered.  
If set to **OFF**:  
• For Microsoft SQL Server 2005 and higher and Microsoft Azure SQL Database, the Hybrid Data Pipeline connectivity service does not report trigger results if the statement is a single INSERT, UPDATE, DELETE, CREATE, ALTER, DROP, GRANT, REVOKE, or DENY statement.  
• For other Microsoft SQL Server databases, the Hybrid Data Pipeline connectivity service does not report trigger results if the statement is a single INSERT, UPDATE, or DELETE statement.  
If set to **OFF**, the only result that is returned is the update count that is generated by the statement that was executed (if no errors occurred). Although trigger results are ignored, any errors and warnings that are generated by the trigger are reported. If errors are reported, the update count is not reported.  
**Default:** OFF |

**Valid values:** **ReadOnly | ReadWrite**  
If set to **ReadOnly**, the Hybrid Data Pipeline connectivity service requests read-only routing and connects to the read-only database replicas as specified by the server.  
If set to **ReadWrite**, the Hybrid Data Pipeline connectivity service connects to a read-write node in the AlwaysOn environment.  
**Default:** ReadWrite |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulk Load Options</td>
<td>Enables bulk load protocol options for batch inserts that the Hybrid Data Pipeline connectivity service can take advantage of when Enable Bulk Load is set to a value of <strong>ON</strong>.</td>
</tr>
</tbody>
</table>

**Valid values:**

0 | 1 | 2 | 16 | 32 | 64

<table>
<thead>
<tr>
<th>Value</th>
<th>Option Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>All of the options are disabled.</td>
</tr>
<tr>
<td>1</td>
<td>The KeepIdentity option preserves identity values. If unspecified, identity values are ignored in the source and are assigned by the destination. Note: If using the bulk load feature with batch inserts, this option has no effect if enabled.</td>
</tr>
<tr>
<td>2</td>
<td>The TableLock option assigns a table lock for the duration of the bulk copy operation. Other applications cannot update the table until the operation completes. If unspecified, the default bulk locking mechanism specified by the database server is used.</td>
</tr>
<tr>
<td>16</td>
<td>The CheckConstraints option checks integrity constraints while data is being copied. If unspecified, constraints are not checked.</td>
</tr>
<tr>
<td>32</td>
<td>The FireTriggers option causes the database server to fire insert triggers for the rows being inserted into the database. If unspecified, triggers are not fired.</td>
</tr>
<tr>
<td>64</td>
<td>The KeepNulls option preserves null values in the destination table regardless of the settings for default values. If unspecified, null values are replaced by column default values where applicable.</td>
</tr>
</tbody>
</table>

**Example**

A value of 67 means the KeepIdentity, TableLock, and KeepNulls options are enabled (1 + 2 + 64).

**Default:** 2
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Catalog Options     | Determines which type of metadata information is included in result sets when a JDBC application calls DatabaseMetaData methods. To include multiple types of metadata information, add the sum of the values that you want to include. In this case, specify 6 to include synonyms and to emulate getColumns() calls.  
**Valid values:**  
2 | 4  
If set to 2, result sets contain synonyms that are returned from the following DatabaseMetaData methods: getColumns(), getExportedKeys(), getFunctionColumns(), getFunctions(), getImportedKeys(), getIndexInfo(), getPrimaryKeys(), getProcedureColumns(), and getProcedures().  
If set to 4, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate getColumns() calls using the ResultSetMetaData object instead of querying database catalogs for column information. Result sets contain synonyms. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumns(). The argument to getColumns() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the Hybrid Data Pipeline connectivity service reverts to the default behavior for getColumns() calls.  
**Default:** 2  

| Code Page Override  | The code page the Hybrid Data Pipeline connectivity service uses to convert Character and Clob data. The specified code page overrides the default database code page or column collation. All Character and Clob data that is returned from or written to the database is converted using the specified code page.  
By default, the Hybrid Data Pipeline connectivity service automatically determines which code page to use to convert Character data. Use this parameter only if you need to change the Hybrid Data Pipeline connectivity service’s default behavior.  
**Valid values:**  
string  
where string is the name of a valid code page that is supported by your JVM. For example, CP950.  
**Default:** empty string |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Enable Bulk Load| Specifies whether to use the bulk load protocol for insert, update, delete, and batch operations. This increases the number of rows that the Hybrid Data Pipeline connectivity service loads to send to the data store. Bulk load reduces the number of network trips.  
**Valid values:**  
ON | OFF  
If set to ON, the Hybrid Data Pipeline connectivity service uses the native bulk load protocols for batch inserts.  
If set to OFF, the Hybrid Data Pipeline connectivity service uses the batch mechanism for batch inserts.  
**Default:** OFF |
| Extended Options| Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
```
Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]
```
If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  
**Valid Values:** string  
**Default:** empty string |
| Initialization String| A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
**Syntax:**  
```
SQLcommand[; SQLcommand]...
```
where:  
`SQLcommand` is a SQL command. Multiple commands must be separated by semicolons.  
The default is an empty string. |
| Load Balancing  | Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property.  
**Valid Values:** ON | OFF  
If set to ON, the connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, to wait for a connection to be established before timing out the connection request.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>[0 \mid x]</td>
</tr>
<tr>
<td></td>
<td>where (x) is a positive integer that represents a number of seconds.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service does not time out a connection request.</td>
</tr>
<tr>
<td></td>
<td>If set to (x), the connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 30</td>
</tr>
<tr>
<td>Max Pooled</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.</td>
</tr>
<tr>
<td>Statements</td>
<td>The default value is 0.</td>
</tr>
<tr>
<td>Multi-Subnet</td>
<td>Determines whether the connectivity service attempts parallel connections to the failover IP addresses of an Availability Group during initial connection or during a multi-subnet failover.</td>
</tr>
<tr>
<td>Failover</td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>[ON \mid OFF]</td>
</tr>
<tr>
<td></td>
<td>If set to ON, the connectivity service will simultaneously attempt to connect to all IP addresses associated with the Availability Group listener when establishing an initial connection or reconnecting after a connection is broken or the listener IP address becomes unavailable. The first IP address to successfully respond to the request is used for the connection. Using parallel-connection attempts offers improved response time over traditional failover, which attempts to connect to alternate servers one at a time.</td>
</tr>
<tr>
<td></td>
<td>If set to OFF, the connectivity service connects to an alternate server or servers as specified by the AlternateServer property when the primary server is unavailable. Use this setting if your environment is not configured for Always On Availability Groups.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Query Timeout</td>
<td>Sets the default query timeout (in seconds) for all statements that are created by a connection.</td>
</tr>
<tr>
<td><strong>Valid values:</strong></td>
<td>-1</td>
</tr>
<tr>
<td>If set to -1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the Statement.setQueryTimeout() method.</td>
<td></td>
</tr>
<tr>
<td>If set to 0, the default query timeout is infinite (the query does not time out).</td>
<td></td>
</tr>
<tr>
<td>If set to x, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection option, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.</td>
<td></td>
</tr>
<tr>
<td>The default value is 0.</td>
<td></td>
</tr>
<tr>
<td>Result Set Meta Data Options</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.</td>
</tr>
<tr>
<td><strong>Valid values:</strong></td>
<td>0</td>
</tr>
<tr>
<td>If set to 0 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getTableName() method may return an empty string for each column in the result set.</td>
<td></td>
</tr>
<tr>
<td>If set to 1 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The Hybrid Data Pipeline connectivity service returns schema name and catalog name information when the ResultSetMetaData.getSchemaName() and ResultSetMetaData.getCatalogName() methods are called if the Hybrid Data Pipeline connectivity service can determine that information.</td>
<td></td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>0</td>
</tr>
</tbody>
</table>
### Table: Select Method

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Select Method**    | A hint to the Hybrid Data Pipeline connectivity service that determines whether the connectivity service requests a database cursor for Select statements. Performance and behavior of the connectivity service are affected by this property, which is defined as a hint because the connectivity service may not always be able to satisfy the requested method. **Valid values:**
  - direct | cursor                                                                                                                                                                                                                                                                       |
  | If set to **direct**, the database server sends the complete result set in a single response to the Hybrid Data Pipeline connectivity service when responding to a query. A server-side database cursor is not created if the requested result set type is a forward-only result set. Typically, responses are not cached by the Hybrid Data Pipeline connectivity service. Using this method, the connectivity service must process the entire response to a query before another query is submitted. If another query is submitted (using a different statement on the same connection, for example), the connectivity service caches the response to the first query before submitting the second query. Typically, the direct method performs better than the cursor method.  
  | If set to **cursor**, a server-side cursor is requested. When returning forward-only result sets, the rows are returned from the server in blocks. The setFetchSize() method can be used to control the number of rows that are returned for each request when forward-only result sets are returned. Performance tests show that, when returning forward-only result sets, the value of Statement.setFetchSize() significantly impacts performance. There is no simple rule for determining the setFetchSize() value that you should use. We recommend that you experiment with different setFetchSize() values to determine which value gives the best performance for your application. The cursor method is useful for queries that produce a large amount of data, particularly if multiple open result sets are used.  |
| **Default:**        | direct                                                                                                                                                                                                                                                                       |

### Table: Snapshot Serializable

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Snapshot Serializable** | For Microsoft SQL Server 2005 and higher and Microsoft Azure SQL Database only. Allows your application to use Snapshot Isolation for connections. This parameter is useful for applications that have the Serializable isolation level set. Using the Snapshot Serializable parameter allows you to use Snapshot Isolation with no or minimum code changes. If you are developing a new application, you may find that using the constant TRANSACTION_SNAPSHOT is a better choice. **Valid values:**
  - ON | OFF  
  | If set to **ON** and your application has the transaction isolation level set to Serializable, the application uses Snapshot Isolation for connections.  
  | If set to **OFF** and your application has the transaction isolation level set to Serializable, the application uses the Serializable isolation level.  |
| **Default:**        | OFF                                                                                                                                                                                                                                                                           |

**Note:** To use Snapshot Isolation, your database also must be configured for Snapshot Isolation.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suppress Connection Warnings</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service suppresses &quot;changed database&quot; and &quot;changed language&quot; warnings when connecting to the database server.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>- ON</td>
</tr>
<tr>
<td></td>
<td>If set to ON, warnings are suppressed.</td>
</tr>
<tr>
<td></td>
<td>If set to OFF, warnings are not suppressed.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong></td>
</tr>
<tr>
<td></td>
<td>OFF</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Transaction Mode    | Specifies how the Hybrid Data Pipeline connectivity service delimits the start of a local transaction.<br><br>**Valid values:** implicit | explicit  
If set to **implicit**, the Hybrid Data Pipeline connectivity service uses implicit transaction mode. This means that the database, not the connectivity service, automatically starts a transaction when a transactionable statement is executed. Typically, implicit transaction mode is more efficient than explicit transaction mode because the connectivity service does not have to send commands to start a transaction and a transaction is not started until it is needed. When TRUNCATE TABLE statements are used with implicit transaction mode, the database may roll back the transaction if an error occurs. If this occurs, use the explicit value for this parameter.<br><br>If set to **explicit**, the Hybrid Data Pipeline connectivity service uses explicit transaction mode. This means that the connectivity service, not the database starts a new transaction if the previous transaction was committed or rolled back.<br><br>**Default:** implicit |
| Metadata Exposed Schemas | Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.<br><br>Warning: This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.<br><br>**Valid Values**<br><br>"<schema>"  
Where:<br><br>"<schema>"  
is the name of a valid schema on the backend data store.<br><br>**Default:** No schema is specified. Therefore, all schemas are exposed. |

**See the steps for:**

How to create a data source in the Web UI on page 225

**MySQL Community Edition parameters**

The following tables describe parameters available on the **General** tab of a MySQL Community Edition Data Source dialog.
Note: Hybrid Data Pipeline uses MySQL Connector/J when connecting to MySQL Community Edition. During installation of the Hybrid Data Pipeline server, you are prompted to specify the location of the MySQL Connector/J driver. Since MySQL Connector/J is a separate component, it may require configuration and maintenance apart from Hybrid Data Pipeline. Therefore, you should refer to MySQL Connector/J documentation for information on support, functionality, and maintenance. In addition, the Progress DataDirect Hybrid Data Pipeline Installation Guide provides a procedure for upgrading the MySQL Connector/J driver without reinstalling the Hybrid Data Pipeline server.

- General tab
- OData tab

General tab

Create MySQL Community Edition Data Source

Table 44: General tab connection parameters for MySQL Community Edition

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials used to connect to the MySQL Community Edition database. A username and password is required if user ID/password authentication is enabled on your database. Contact your system administrator to obtain your user name.</td>
</tr>
<tr>
<td>Note:</td>
<td>By default, the password is encrypted.</td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Server Name</td>
<td>Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, 122.23.15.12 or mysqlcommunityserver.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The TCP port of the primary database server listening for connections to the database.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example: Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;] If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence. <strong>Valid Values:</strong> string <strong>Default:</strong> empty string</td>
</tr>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
</tbody>
</table>
OData tab

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.

Table 45: OData tab connection parameters for MySQL Community Edition

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
### Field: OData Access URI

Specifies the base URI for the OData feed to access your data source, for example, `https://hybridpipe.operations.com/api/odata/<DataSourceName>`. You can copy the URI and paste it into your application's OData configuration.

The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding `/$metadata` to the service root URI.

### Field: Schema Map

Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.

See [Configuring data sources for OData connectivity and working with data source groups](on page 622) for more information.

### Field: Data Source Caching

Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

**Valid Values:**

- When set to 1, session caching is enabled. This provides better performance for production.
- When set to 0, session caching is disabled. Use this value when you are configuring the data source.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Page Size</strong></td>
<td>Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the <code>$top</code> and <code>$skip</code> parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.</td>
</tr>
</tbody>
</table>
|                  | **Valid Values:** 0 | $n$  
|                  | where $n$ is an integer from 1 to 10000.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|                  | When set to 0, the server default of 2000 is used.                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|                  | **Default:** 0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| **Refresh Result** | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change. |
|                  | **Valid Values:**  
|                  | When set to 0, the OData service caches the first page of results.                                                                                                                                                                                                                                                                                                                                                                                         |
|                  | When set to 1, the OData service re-executes the query.                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|                  | **Default:** 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| **Inline Count Mode** | Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging. |
|                  | The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large. |
|                  | **Valid Values:**  
<p>|                  | When set to 1, the connectivity service runs a separate <code>count(*)</code> aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.                                                                                                                                                                                                 |
|                  | When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.                                                                                                                                                                                                                     |
|                  | <strong>Default:</strong> 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client-side pagination, allowing the service to better anticipate how to process queries. Valid Values: Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip. Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result. Default: 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled. Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon. Valid Values: ON</td>
</tr>
<tr>
<td></td>
<td>When ON is selected, OData access is restricted to read-only mode. When OFF is selected, write operations can be performed on the OData service. Default: OFF</td>
</tr>
</tbody>
</table>

**MySQL Enterprise parameters**

The following tables describe parameters available on the tabs of a MySQL Data Source dialog:

- General Tab
- OData tab
- Security tab
- Advanced tab
General tab

Create MySQL Data Source

<table>
<thead>
<tr>
<th>General</th>
<th>Security</th>
<th>OData</th>
<th>Advanced</th>
</tr>
</thead>
</table>

- **Data Source Name**: 
- **Description**: 
- **User ID**: 
- **Password**: 
- **Server Name**: 
- **Port Number**: 3306
- **Database**: 
- **Connector ID**: 

*Required Fields*
Table 46: General tab connection parameters for MySQL Enterprise

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The TCP port of the primary database server listening for connections to the database.</td>
</tr>
<tr>
<td>Server Name</td>
<td>Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, 122.23.15.12 or mysqlserver.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials used to connect to the MySQL database. A user name and password is required if user ID/password authentication is enabled on your database. Contact your system administrator to obtain your user name. Note: By default, the password is encrypted. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
</tbody>
</table>
Security tab

Create MySQL Data Source

Table 47: Security tab connection parameters for MySQL Enterprise

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Crypto Protocol Version | Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.  

**Valid Values:**

cryptographic_protocol [[, cryptographic_protocol]...]

where:

cryptographic_protocol

is one of the following cryptographic protocols:

TLSv1 | TLSv1.1 | TLSv1.2

The client must send the highest version that it supports in the client hello.

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.

**Example**

Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

TLSv1.1, TLSv1.2

**Default:** TLSv1, TLSv1.1, TLSv1.2
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption Method</td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server.</td>
</tr>
</tbody>
</table>

**Valid Values:**

- noEncryption | SSL

If set to noEncryption, data is not encrypted or decrypted.

If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.

**Note:**

- Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the Login Timeout parameter to avoid problems when connecting to a server that does not support SSL.

- When SSL is enabled, the following properties also apply:
  
  - Host Name In Certificate
  - ValidateServerCertificate
  - Crypto Protocol Version

The default value is noEncryption.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Host Name In Certificate | Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.  

**Valid Values:**  

`host_name | #SERVERNAME#`

where `host_name` is a valid host name.

If `host_name` is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.

If `#SERVERNAME#` is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.

**Default:** Empty string

<p>| Validate Server Certificate | Description |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.</td>
</tr>
</tbody>
</table>

**Valid Values:**

ON | OFF

If set to ON, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate parameter is specified, the connectivity service also validates the certificate using a host name. The Host Name In Certificate parameter is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the connectivity service is connecting to is the server that was requested.

If set to OFF, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any truststore information that is specified by the Java system properties. Truststore information is specified using Java system properties.

**Default:** ON

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 48: OData tab connection parameters for MySQL Enterprise

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
| OData Access URI | Specifies the base URI for the OData feed to access your data source, for example, `https://hybridpipe.operations.com/api/odata/<DataSourceName>`. You can copy the URI and paste it into your application's OData configuration.  
The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.  
The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding `/$metadata` to the service root URI. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Page Size</td>
<td>Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages. <strong>Valid Values:</strong> 0</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Refresh Result| Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1 |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
**Valid Values:**  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
**Default:** 1 |
### Top Mode

Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**
- Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.
- Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.

**Default:** 0

### OData Read Only

Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.

Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.

**Valid Values:**
- ON | OFF

  - When **ON** is selected, OData access is restricted to read-only mode.
  - When **OFF** is selected, write operations can be performed on the OData service.

**Default:** OFF
Advanced tab

Table 49: Advanced tab connection parameters for MySQL Enterprise

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alternate Servers</td>
<td>Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property. <strong>Valid Values:</strong>&lt;br&gt;$(servername1[:port1],servername2[:port2])...$&lt;br&gt;The server name $(servername1, servername2, and so on)$ is required for each alternate server entry. Port number $(port1, port2, and so on)$ is optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number is used. <strong>Default:</strong> None</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Catalog Options</td>
<td>Determines which type of metadata information is included in result sets when a JDBC application calls DatabaseMetaData methods.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>If set to 2, result sets contain synonyms that are returned from the following DatabaseMetaData methods: getColumn(), getExportedKeys(), getFunctionColumns(), getFunctions(), getImportedKeys(), getIndexInfo(), getPrimaryKeys(), getProcedureColumns(), and getProcedures().</td>
</tr>
<tr>
<td></td>
<td>If set to 4, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate getColumn() calls using the ResultSetMetaData object instead of querying database catalogs for column information. Result sets contain synonyms. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumn(). The argument to getColumn() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the connectivity service reverts to the default behavior for getColumn() calls.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 2</td>
</tr>
<tr>
<td>Code Page Override</td>
<td>The code page to be used by the Hybrid Data Pipeline connectivity service to convert Character and Clob data. The specified code page overrides the default database code page or column collation. All Character and Clob data that is returned from or written to the database is converted using the specified code page.</td>
</tr>
<tr>
<td></td>
<td>By default, the Hybrid Data Pipeline connectivity service automatically determines which code page to use to convert Character data. Use this parameter only if you need to change the Hybrid Data Pipeline connectivity service’s default behavior.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>string</td>
</tr>
<tr>
<td></td>
<td>where string is the name of a valid code page that is supported by your JVM. For example, CP950.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> empty string</td>
</tr>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:</td>
</tr>
<tr>
<td></td>
<td>Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]</td>
</tr>
<tr>
<td></td>
<td>If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> string</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> empty string</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Initialization String | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
**Syntax:**

```
command[[; command]...]
```

**Where:**

- `command`

   is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:

```
InitializationString=(REFRESH SCHEMA SFORCE)
```

The default is an empty string. |
| Load Balancing | Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property.  
**Valid Values:** ON | OFF  
If set to ON, the connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established.  
If set to OFF, the connectivity service does not use client load balancing and connects to each servers based on their sequential order (primary server first, then, alternate servers in the order they are specified).  
**Default:** OFF  
**Notes**

- The Alternate Servers connection parameter specifies one or multiple alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers parameter. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, to wait for a connection to be established before timing out the connection request.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>where x is a positive integer that represents a number of seconds.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service does not time out a connection request.</td>
</tr>
<tr>
<td></td>
<td>If set to x, the connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 30</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.</td>
</tr>
<tr>
<td></td>
<td>The default value is 0.</td>
</tr>
</tbody>
</table>
### Query Timeout

Sets the default query timeout (in seconds) for all statements that are created by a connection.

**Valid Values:**

- `-1 | 0 | x`

If set to `-1`, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the `Statement.setQueryTimeout()` method.

If set to `0`, the default query timeout is infinite (the query does not time out).

If set to `x`, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection option, call the `Statement.setQueryTimeout()` method to set a timeout value for a particular statement.

The default value is 0.

### ResultSet Meta Data Options

Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.

**Valid Values:**

- `0 | 1`

If set to `0` and the `ResultSetMetaData.getTableName()` method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The `getTableName()` method may return an empty string for each column in the result set.

If set to `1` and the `ResultSetMetaData.getTableName()` method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the `ResultSetMetaData.getSchemaName()` and `ResultSetMetaData.getCatalogName()` methods are called if the connectivity service can determine that information.

**Default:** 0

---

**See the steps for:**

*How to create a data source in the Web UI* on page 225

---

### Oracle parameters

The following tables describe parameters available on the tabs of the Oracle **Data Source** dialog:

- **General tab**
- **OData tab**
- **Security tab**
- **Advanced tab**
General tab

Create Oracle Data Source

- Data Source Name
- Description
- User ID
- Password
- Server Name
- Port Number
- Connector ID
- Edition Name
- Service Name
- SID
- SysLoginRole
- TNS Names File
- TNS Server Name

Required Fields
### Table 50: General tab connection parameters for Oracle

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>User Id</td>
<td>The User Id for the Oracle account used to establish the connection to the Oracle server.</td>
</tr>
<tr>
<td>Password</td>
<td>A password for the Oracle account that is used to establish the connection to your Oracle server.</td>
</tr>
<tr>
<td>Note:</td>
<td>By default, the password is encrypted. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Server Name</td>
<td>Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, 122.23.15.12 or OracleAppServer. If using a tnsnames.ora file to provide connection information, do not specify this parameter. Valid values: string where: string is a valid IP address or server name. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The port number on which the Oracle database instance is listening for connections.</td>
</tr>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Edition Name  | The name of the Oracle edition the Hybrid Data Pipeline connectivity service uses when establishing a connection. Oracle 11gR2 and higher allows your database administrator to create multiple editions of schema objects so that your application can still use those objects while the database is being upgraded. This parameter is only valid for Oracle 11g R2 and higher databases and tells the connectivity service which edition of the schema objects to use. The Hybrid Data Pipeline connectivity service uses the default edition in the following cases:  
  • When the specified edition is not a valid edition. The Hybrid Data Pipeline connectivity service generates a warning indicating that it was unable to set the current edition to the specified edition.  
  • When the value for this parameter is not specified or is set to an empty string.  
  **Valid values:**  
  ```  
  string  
  ```  
  where:  
  ```  
  string  
  ```  
  is the name of a valid Oracle edition.  
  **Default:** empty string |
<p>| Service Name  | The Oracle Service Name that identifies the database on the Oracle server to connect to.                                                                                                                      |
| SID           | The Oracle SID that identifies the database on the Oracle server to connect to.                                                                                                                             |
|               | <strong>Note:</strong> Oracle recommends using Oracle Server Name instead of SID.                                                                                                                                       |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| SysLoginRole| Specifies whether the user is logged on the database with the Oracle system privilege SYSDBA or the Oracle system privilege SYSOPER. For example, you may want the user to be granted the SYSDBA privilege to allow the user to create or drop a database. Refer to your Oracle documentation for information about which operations are authorized for the SYSDBA and SYSOPER system privileges. **Valid values:**  
  sysdba | sysoper  

  If set to `sysdba`, the user is logged on the database with the Oracle system privilege SYSDBA. The user must be granted SYSDBA system privileges before the connection is attempted by the Hybrid Data Pipeline connectivity service. If not, the connectivity service returns an error and the connection attempt fails. 

  If set to `sysoper`, the user is logged on the database with the Oracle system privilege SYSOPER. The user must be granted SYSOPER system privileges before the connection is attempted by the Hybrid Data Pipeline connectivity service. If not, the connectivity service throws an exception and the connection attempt fails. 

  If this parameter is set to an empty string or is unspecified, the user is logged in without SYSDBA or SYSOPER privileges.  

  **Default:** empty string
### TNS Names File

Specifies the name of the TNSNAMES.ORA file. In a TNSNAMES.ORA file, connection information for Oracle services is associated with an Oracle net service name. The entry in the TNSNAMES.ORA file specifies Host, Port Number, and Service Name or SID.

TNSNames File is ignored if no value is specified in the Server Name option. If the Oracle Server Name option is specified but the TNSNames File option is left blank, the TNS_ADMIN environment setting is used for the TNSNAMES.ORA file path. If there is no TNS_ADMIN setting, the ORACLE_HOME environment setting is used. On Windows, if ORACLE_HOME is not set, the path is taken from the Oracle section of the Registry.

Using an Oracle TNSNAMES.ORA file to centralize connection information in your Oracle environment simplifies maintenance when changes occur. If, however, the TNSNAMES.ORA file is unavailable, then it is useful to be able to open a backup version of the TNSNAMES.ORA file (TNSNames file failover). You can specify one or more backup, or alternate, TNSNAMES.ORA files.

**Valid values:**

```
path_filename
```

where:

```
path_filename
```

is the entire path, including the file name, to the TNSNAMES.ORA file.

To specify multiple TNSNAMES.ORA file locations, separate the names with a comma and enclose the locations in parentheses (you do not need parentheses for a single entry). For example:

```
(M:\server2\oracle\tnsnames.ora,
C:\oracle\product\10.1\db_1\network\admin\tnsnames.ora)
```

The Hybrid Data Pipeline connectivity service tries to open the first file in the list. If that file is not available, then it tries to open the second file in the list, and so on.

**Note:** This option is mutually exclusive with the Server Name, Port Number, SID, and Service Name options.

### TNS Server Name

Specifies the name of the set of connection information in the tnsnames.ora file to use to establish the connection.
Creating data sources with the Web UI

Security tab

Create Oracle Data Source

<table>
<thead>
<tr>
<th>General</th>
<th>Security</th>
<th>OData</th>
<th>Advanced</th>
</tr>
</thead>
</table>

Data Integrity Level
- accepted

Data Integrity Types
- (MD5,SHA1,SHA512,SHA384,SHA512)

Encryption Level
- accepted

Encryption Method
- none

Encryption Types
- (3DES112, 3DES168, AES128, AES256, DES, RC4_40, RC4_56)

Crypto Protocol Version
- TLSv1, TLSv1.1, TLSv1.2

Host Name in Certificate

Validate Server Certificate
- ON
### Table 51: Security tab connection parameters for Oracle

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Data Integrity Level   | Determines the level of Oracle Advanced Security data integrity used for data sent between the Hybrid Data Pipeline connectivity service and database server. The connection fails if the database server does not have a compatible integrity algorithm.  
**Valid values:**  
rejected | accepted | requested | required  
If set to rejected, the Hybrid Data Pipeline connectivity service does not enable a data integrity check for data sent between the connectivity service and database server. The connection fails if the database server specifies REQUIRED.  
If set to accepted, the Hybrid Data Pipeline connectivity service enables a data integrity check for data sent between the connectivity service and database server if the database server requests or requires it.  
If set to requested, the Hybrid Data Pipeline connectivity service enables a data integrity check for data sent between the connectivity service and database server if the database server permits it.  
If set to required, the Hybrid Data Pipeline connectivity service performs a data integrity check for data sent between the connectivity service and database server. The database server must have data integrity check enabled. The connection fails if the database server specifies REJECTED.  
**Note:**  
- You can enable data integrity protection without enabling encryption.  
- Consult your Oracle administrator to verify the data integrity settings of your Oracle server.  
**Default:** accepted
Determines the method the Hybrid Data Pipeline connectivity service uses to protect against attacks that intercept and modify data being transmitted between the client and server. You can enable data integrity protection without enabling encryption.

Valid values:

value [(,value ]...)

where:

value

is one of the following values specifying an algorithm in the following table:

Table 52: Oracle Advanced Security data integrity algorithms

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD5</td>
<td>Message Digest 5 (MD5).</td>
</tr>
<tr>
<td>SHA1</td>
<td>Secure Hash Algorithm (SHA-1).</td>
</tr>
</tbody>
</table>

Note:

- Multiple values must be separated by commas. In addition, if this parameter is specified in a connection URL, the entire value must be enclosed in parentheses when multiple values are specified.
- If multiple values are specified and Oracle Advanced Security data integrity is enabled using the Data Integrity Level parameter, the database server determines which algorithm is used based on how it is configured.
- If unspecified, a list of all possible values is sent to the database server. The database server determines which algorithm is used based on how it is configured.
- The value of this parameter is ignored if the Data Integrity Level parameter is set to rejected.
- Consult your Oracle administrator to verify the data encryption settings of your Oracle server.

Default: SHA1,MD5 (a list of all possible values)
### Encryption Level

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and database server using Oracle Advanced Security encryption.</td>
</tr>
<tr>
<td><strong>Valid values:</strong></td>
<td></td>
</tr>
<tr>
<td>rejected</td>
<td></td>
</tr>
<tr>
<td>accepted</td>
<td></td>
</tr>
<tr>
<td>requested</td>
<td></td>
</tr>
<tr>
<td>required</td>
<td></td>
</tr>
</tbody>
</table>

If set to **rejected**, data sent between the Hybrid Data Pipeline connectivity service and the database server is not encrypted or decrypted. The connection fails if the database server specifies **REQUIRED**.

If set to **accepted**, data sent between the Hybrid Data Pipeline connectivity service and the database server is encrypted and decrypted if the database server requests or requires it.

If set to **requested**, data sent between the Hybrid Data Pipeline connectivity service and the database server is encrypted and decrypted if the database server permits it.

If set to **required**, data sent between the Hybrid Data Pipeline connectivity service and the database server must be encrypted and decrypted. The connection fails if the database server specifies **REJECTED**.

**Note:**

- When this parameter is set to **accepted**, **requested**, or **required**, the Encryption Types connection parameter determines which Oracle Advanced Security algorithms are used.
- To enable SSL encryption, you can set the Encryption Method connection parameter.
- Consult your database administrator to verify the data encryption settings of your Oracle server.

**Default:** **accepted**
Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server.

**Valid values:**

- **noEncryption**  
- **SSL**

If set to **noEncryption**, data is not encrypted or decrypted.

If set to **SSL**, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.

**Note:**

- Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the Login Timeout parameter to avoid problems when connecting to a server that does not support SSL.

- When SSL is enabled, the following properties also apply:
  
  - **HostNameInCertificate**
  - **ValidateServerCertificate**
  - **CryptoProtocolVersion**

- To enable Oracle Advanced Security encryption, you can set the Encryption Level connection parameter.

The default value is **noEncryption**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Encryption Method</strong></td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server.</td>
</tr>
</tbody>
</table>
| **Valid values:**      | **noEncryption**  
|                        | **SSL**  |
| **If set to noEncryption**, data is not encrypted or decrypted.**  |
| **If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.**  |
| **Note:**              |  
|                        | • Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the Login Timeout parameter to avoid problems when connecting to a server that does not support SSL.  
|                        | • When SSL is enabled, the following properties also apply:  
|                        |   - **HostNameInCertificate**  
|                        |   - **ValidateServerCertificate**  
|                        |   - **CryptoProtocolVersion**  
<p>|                        | • To enable Oracle Advanced Security encryption, you can set the Encryption Level connection parameter.  |
| <strong>The default value is noEncryption.</strong> |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption Types</td>
<td></td>
</tr>
</tbody>
</table>
Specifies a comma-separated list of the encryption algorithms to use if Oracle Advanced Security encryption is enabled using the Encryption Level parameter.

**Valid values:**

```plaintext
encryption_algorithm [[,encryption_algorithm ]...]  
```

```plaintext
effort=algorithm
```

**Encryption algorithm**

<table>
<thead>
<tr>
<th>Encryption algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DES112</td>
<td>Two-key Triple-DES (with an effective key size of 112-bit).</td>
</tr>
<tr>
<td>AES128</td>
<td>AES with a 128-bit key size.</td>
</tr>
<tr>
<td>AES192</td>
<td>AES with a 192-bit key size.</td>
</tr>
<tr>
<td>AES256</td>
<td>AES with a 256-bit key size.</td>
</tr>
<tr>
<td>DES</td>
<td>DES (with an effective key size of 56-bit).</td>
</tr>
<tr>
<td>DES168</td>
<td>Three-key Triple-DES (with an effective key size of 168-bit).</td>
</tr>
<tr>
<td>RC4_128</td>
<td>RC4-128 with a 128-bit key size.</td>
</tr>
<tr>
<td>RC4_256</td>
<td>RC4 with a 256-bit key size.</td>
</tr>
<tr>
<td>RC4_40</td>
<td>RSA RC4 with a 40-bit key size.</td>
</tr>
<tr>
<td>RC4_56</td>
<td>RSA RC4 with a 56-bit key size.</td>
</tr>
</tbody>
</table>

**Note:** Beginning with Oracle 11.2, Oracle no longer supports DES, MD5, and RC4.

**Example**

Your security environments specifies that you can use AES with a 192-bit key size or two-key Triple-DES with an effective key size of 112-bit. Use the following values:

```
Encryption Types=AES192,3DES112
```

**Note:**

- Multiple values must be separated by commas. In addition, if this parameter is specified in a connection URL, the entire value must be enclosed in parentheses when multiple values are specified.
If multiple values are specified and Oracle Advanced Security encryption is enabled using the Encryption Level parameter, the database server determines which algorithm is used based on how it is configured.

- If unspecified, a list of all possible values is sent to the database server. The database server determines which algorithm is used based on how it is configured.
- Consult your Oracle administrator to verify the data encryption settings of your Oracle server.
- The value of this property is ignored if the Encryption Level parameter is set to rejected.

The default value is an empty string.

**Crypto Protocol Version**

Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.

**Valid Values:**

```
cryptographic_protocol [, cryptographic_protocol]...
```

where:

- `cryptographic_protocol` is one of the following cryptographic protocols:
  - TLSv1 | TLSv1.1 | TLSv1.2
  - The client must send the highest version that it supports in the client hello.

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.

**Example**

Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

```
TLSv1.1, TLSv1.2
```

**Default:** TLSv1, TLSv1.1, TLSv1.2
Specifications

### Host Name In Certificate

Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.

**Valid values:**

- `host_name` | `#SERVERNAME#`

  where `host_name` is a valid host name.

  If `host_name` is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.

  If `#SERVERNAME#` is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.

  **Default:** Empty string

### Validate Server Certificate

Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

**Valid values:**

- `ON` | `OFF`

  If set to `ON`, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate parameter is specified, the connectivity service also validates the certificate using a host name. The Host Name In Certificate parameter is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the connectivity service is connecting to is the server that was requested.

  If set to `OFF`, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any truststore information that is specified by the Java system properties.

  **Default:** `ON`
**OData tab**

The following table describes the controls on the **OData** tab. For information on using the **Configure Schema** editor, see [Configuring data sources for OData connectivity and working with data source groups](#) on page 622. For information on formulating OData requests, see "Formulating queries" under [Querying with OData](#).

![Create Oracle Data Source](image)

**Table 53: OData tab connection parameters for Oracle**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.

### Field: OData Access URI

Specifies the base URI for the OData feed to access your data source, for example, https://hybridpipe.operations.com/api/odata/<DataSourceName>. You can copy the URI and paste it into your application's OData configuration.

### Field: Schema Map

Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this Data Source definition. Use the Configure Schema editor to select the tables/columns and/or functions to expose through OData.

### Field: Data Source Caching

Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

**Valid Values:**

- When set to 1, session caching is enabled. This provides better performance for production.
- When set to 0, session caching is disabled. Use this value when you are configuring the data source.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Page Size           | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages. **Valid Values:** 0 | n  
|                     | where n is an integer from 1 to 10000.  
|                     | When set to 0, the server default of 2000 is used.  
|                     | **Default:** 0  
| Refresh Result      | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change. **Valid Values:**  
|                     | When set to 0, the OData service caches the first page of results.  
|                     | When set to 1, the OData service re-executes the query.  
|                     | **Default:** 1  
| Inline Count Mode   | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
|                     | The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large. **Valid Values:**  
|                     | When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
|                     | When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
<p>|                     | <strong>Default:</strong> 1 |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Top Mode            | Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.  
**Valid Values:**  
Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.  
Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.  
**Default:** 0 |
| OData Read Only     | Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.  
Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.  
**Valid Values:**  
ON | OFF  
When ON is selected, OData access is restricted to read-only mode.  
When OFF is selected, write operations can be performed on the OData service.  
**Default:** OFF |
Advanced tab
Creating data sources with the Web UI
### Table 54: Advanced tab connection parameters for Oracle

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Alternate Servers</strong></td>
<td>Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property.</td>
</tr>
<tr>
<td><strong>Valid values:</strong></td>
<td>((\text{servername}_1[:\text{port}_1][;\text{property}=\text{value}][;...]][,\text{servername}_2[:\text{port}_2][;\text{property}=\text{value}][;...]][,...]))</td>
</tr>
<tr>
<td></td>
<td>The server name ((\text{servername}_1, \text{servername}_2, \text{and so on})) is required for each alternate server entry. Port number ((\text{port}_1, \text{port}_2, \text{and so on})) and connection properties ((\text{property}=\text{value})) are optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number of 1521 is used.</td>
</tr>
<tr>
<td></td>
<td>Optional connection properties are Service Name and SID.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>\text{Server Name=}server1:1521;ServiceName=TEST;AlternateServers=(server2:1521;ServiceName=TEST2,server3:1521;ServiceName=TEST3}</td>
</tr>
<tr>
<td><strong>Bulk Load Options</strong></td>
<td>Enables bulk load protocol options for batch inserts that the Hybrid Data Pipeline connectivity service can take advantage of when EnableBulkLoad is set to a value of ON.</td>
</tr>
<tr>
<td></td>
<td>This option only applies to connections to Oracle 11g R2 and higher database servers.</td>
</tr>
<tr>
<td><strong>Valid values:</strong></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>If set to 0 or unspecified, the bulk load operation continues even if a value that can cause an index to be invalidated is loaded.</td>
</tr>
<tr>
<td></td>
<td>If set to 128, the NoIndexErrors option stops a bulk load operation when a value that would cause an index to be invalidated is loaded. For example, if a value is loaded that violates a unique or non-null constraint, the Hybrid Data Pipeline connectivity service stops the bulk load operation and discards all data being loaded, including any data that was loaded prior to the problem value.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>0</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Catalog Options | Determines which type of metadata information is included in result sets when a JDBC application calls DatabaseMetaData methods.  
Valid values:

| 0 | 1 | 2 | 3 | 4 | 6 | 8 | 10 |

If set to 0, result sets do not contain remarks or synonyms.
If set to 1, result sets contain remarks information that is returned from the following DatabaseMetaData methods: getColumns() and getTables().
If set to 2, result sets contain synonyms that are returned from the following DatabaseMetaData methods: getColumns(), getExportedKeys(), getFunctionColumns(), getFunctions(), getImportedKeys(), getIndexInfo(), getPrimaryKeys(), getProcedureColumns(), and get Procedures().
If set to 3, result sets contain both remarks and synonyms (as described for values 1 and 2).
If set to 4 or 6, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate getColumns() calls using the ResultSetMetaData object instead of querying database catalogs for column information. Result sets contain synonyms, but no remarks. If set to 4, synonyms are not returned for getColumns() calls and getTables() or getProcedure() calls. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumns(). The argument to getColumns() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the Hybrid Data Pipeline connectivity service reverts to the default behavior for getColumns() calls.
If set to 8, result sets contain accurate metadata information for VARRAY, TABLE, and OBJECT data when the following DatabaseMetaData methods are called: getColumns(), getProcedureColumns(), and getFunctionColumns(). Setting this value can negatively impact performance.
If set to 10, results sets contain accurate metadata information for VARRAY, TABLE, and OBJECT data (as described for value 8) and synonyms for other data types (as described for value 2).
Default: 2
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Page Override</td>
<td></td>
</tr>
</tbody>
</table>
The code page to be used by the Hybrid Data Pipeline connectivity service to convert Character data. The specified code page overrides the default database code page or column collation. All Character data that is returned from or written to the database is converted using the specified code page. This option has no effect on how the Hybrid Data Pipeline connectivity service converts character data to the national character set.

By default, the Hybrid Data Pipeline connectivity service automatically determines which code page to use to convert Character data. Use this parameter only if you need to change the connectivity service’s default behavior.

**Valid values:**

- utf8
- sjis
- enhanced_sjis
- enhanced_sjis_oracle
- ms932
- euc jp solaris

where `string` is the name of a valid code page that is supported by your JVM. For example, `CP950`.

If set to `utf8`, the Hybrid Data Pipeline connectivity service uses the UTF-8 code page to send data to the Oracle server as Unicode. The UTF-8 code page converts data from the Java String format UTF-16 to UTF-8. If you specify this value, the Hybrid Data Pipeline connectivity service forces the value of the WireProtocolMode parameter to 2.

If set to `sjis`, the Hybrid Data Pipeline connectivity service uses the SHIFT-JIS code page to convert character data to the JA16SJIS character set.

If set to `enhanced_sjis`, the Hybrid Data Pipeline connectivity service uses the ENHANCED_SJIS code page to convert character data from the Java String format UTF-16 to SJIS as defined by the ICU character conversion library. In addition, it maps the following MS-932 characters to the corresponding SJIS encoding for those characters:

| \UFF5E | Wave dash |
| \U2225 | Double vertical line |
| \UFFE0 | Cent sign |
| \UFF0D | Minus sign |
| \UFFE1 | Pound sign |
| \UFFE2 | Not sign |

This value is provided for backward compatibility. Only use this value when the Oracle database character set is SHIFT_JIS.

If set to `enhanced_sjis_oracle`, the Hybrid Data Pipeline connectivity service uses the ENHANCED_SJIS_ORACLE code page to convert Character data from the Java String format UTF-16 to Oracle's definition of SJIS. When the connectivity service connects to an Oracle database with a JA16SJIS character set, the Hybrid Data Pipeline connectivity service uses this code page by default. The ENHANCED_SJIS_ORACLE code page is a super set of the MS932 code page. Only use this value when the Oracle database character set is SHIFT_JIS.

If set to `ms932`, the Hybrid Data Pipeline connectivity service uses the Microsoft MS932 code page to convert Character data from the Java String format UTF-16 to SJIS. This
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value is provided for backward compatibility because earlier versions of the connectivity service used the MS932 code page when converting Character data to JA16SJIS. Only use this value when the Oracle database character set is SHIFT_JIS. If set to euc_jp_solaris, the Hybrid Data Pipeline connectivity service uses the EUC_JP_Solaris code page to convert Character data to the EUC_JP character set. Default: empty string</td>
<td></td>
</tr>
</tbody>
</table>
| Enable Bulk Load             | Specifies whether to use the bulk load protocol for insert, update, delete, and batch operations. This increases the number of rows that the Hybrid Data Pipeline connectivity service loads to send to the data store. Bulk load reduces the number of network trips. **Valid values:**  
  ON | OFF  
  If set to **ON**, the Hybrid Data Pipeline connectivity service uses the native bulk load protocols for batch inserts. If set to **OFF**, the connectivity service uses the batch mechanism for batch inserts. Default: OFF                                                                                                                                                                                                                                                                                                                                                                                   |
| Extended Options             | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
  `Database=Server1; UndocumentedOption1=value[; UndocumentedOption2=value;]`  
  If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence. **Valid Values:** string  
  Default: empty string                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Fetch TSWTZ AsTimestamp      | Determines whether column values with the TIMESTAMP WITH TIME ZONE data type are returned as a JDBC CHAR or TIMESTAMP data type. Valid on Oracle 10g R2 or higher. **Valid values:**  
  ON | OFF  
  If set to **ON**, column values with the TIMESTAMP WITH TIME ZONE data type are returned as a JDBC TIMESTAMP data type. If set to **OFF**, column values with the TIMESTAMP WITH TIME ZONE data type are returned as a JDBC VARCHAR data type. Default: OFF                                                                                                                                                                                                                                                                                                                                                                    |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Initialization String | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
**Syntax:**  
`SQLcommand[; SQLcommand]...`  
where:  
`SQLcommand` is a SQL command. Multiple commands must be separated by semicolons.  
**Default:** empty string |
| Load Balancing      | Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property.  
**Valid Values:** ON | OFF  
If set to **ON**, the connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established.  
If set to **OFF**, the connectivity service does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).  
**Default:** OFF  
**Notes**  
- The Alternate Servers parameter specifies one or multiple alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers property. |
| Login Timeout       | The amount of time, in seconds, that the Hybrid Data Pipeline connectivity service waits for a connection to be established before timing out the connection request.  
**Valid values:**  
`0 | x`  
where `x` is a positive integer that represents a number of seconds.  
If set to 0, the Hybrid Data Pipeline connectivity service does not time out a connection request.  
If set to `x`, the Hybrid Data Pipeline connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.  
**Default:** 30 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the Hybrid Data Pipeline connectivity service caches the last 20 prepared statements that are created by the application. The default value is 0.</td>
</tr>
<tr>
<td>Query Timeout</td>
<td>Sets the default query timeout (in seconds) for all statements created by a connection.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>where x is a positive integer that represents a number of seconds.</td>
</tr>
<tr>
<td></td>
<td>If set to -1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the Statement.setQueryTimeout() method.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the default query timeout is infinite (the query does not time out).</td>
</tr>
<tr>
<td></td>
<td>If set to x, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value that is set by this parameter, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>Report Recycle Bin</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service returns items that are in the Oracle Recycle Bin for the getTables(), getColumns(), and getTablePrivileges() methods. For Oracle 10g R1 and higher, when a table is dropped, it is not actually removed from the database, but is placed in the recycle bin. By default, the connectivity service returns items in the Oracle Recycle Bin.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid values:</strong></td>
</tr>
<tr>
<td></td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>If set to ON, the Hybrid Data Pipeline connectivity service fetches items contained in the Oracle Recycle Bin.</td>
</tr>
<tr>
<td></td>
<td>If set to OFF, the Hybrid Data Pipeline connectivity service does not return items contained in the Oracle Recycle Bin. Functionally, this means that the Hybrid Data Pipeline connectivity service filters out results whose table name begins with BIN$.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> ON</td>
</tr>
</tbody>
</table>
### Resultset Meta Data Options

Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.

**Valid values:**

0 | 1

If set to 0 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getTableName() method may return an empty string for each column in the result set.

If set to 1 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the ResultSetMetaData.getSchemaName() and ResultSetMetaData.getCatalogName() methods are called if the connectivity service can determine that information.

**Default:** 0

### Send Float Parameters As String

Determines whether FLOAT, BINARY_FLOAT, and BINARY_DOUBLE parameters are sent to the database server as a string or as a floating point number.

**Valid values:**

ON | OFF

If set to ON, the Hybrid Data Pipeline connectivity service sends FLOAT, BINARY_FLOAT, and BINARY_DOUBLE parameters to the database server as string values.

If set to OFF, the Hybrid Data Pipeline connectivity service sends FLOAT, BINARY_FLOAT, and BINARY_DOUBLE parameters to the database server as floating point numbers. When Oracle overloaded stored procedures are used, this value ensures that the database server can determine the correct stored procedure to call based on the parameter’s data type.

**Note:**

- Numbers larger than 1.0E127 or smaller than 1.0E-130 cannot be converted to Oracle’s number format for Oracle8i and Oracle9i databases using floating point numbers. When a number larger than 1.0E127 or smaller than 1.0E-130 is encountered, the Hybrid Data Pipeline connectivity service throws an exception. If your application uses numbers in this range against an Oracle8i or Oracle9i database, set this parameter to ON.

**Default:** OFF
Determined whether the Hybrid Data Pipeline connectivity service uses ORA_CHAR or ORA_VARCHAR bindings for string parameters in a Where clause. Using ORA_VARCHAR bindings can improve performance, but may cause matching problems for CHAR columns.

**Valid values:**

| ON | OFF |

If set to **ON**, the Hybrid Data Pipeline connectivity service uses ORA_CHAR bindings. If set to **OFF**, the Hybrid Data Pipeline connectivity service uses ORA_VARCHAR bindings, which can improve performance. For example, in the following code, if col1 is defined as CHAR(10) and the column name has the string 'abc' in it, the match will fail.

```java
ps = con.prepareStatement("SELECT * FROM employees WHERE col1=?");
ps.setString(1, "abc");
rs = ps.executeQuery();
```

**Default:** ON
Support Links

Determines whether the Hybrid Data Pipeline connectivity service supports Oracle linked servers, which means a mapping has been defined in one Oracle server to another Oracle server. When Oracle linked servers are supported, the connectivity service does not support distributed transactions.

Valid values:

- **ON**
- **OFF**

If set to **ON**, the Hybrid Data Pipeline connectivity service supports Oracle linked servers but does not support distributed transactions.

If set to **OFF**, the Hybrid Data Pipeline connectivity service supports distributed transactions but does not support Oracle linked servers. In most cases, setting this parameter to **OFF** provides the best performance.

Default: **OFF**

Metadata Exposed Schemas

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

⚠️ **Warning**: This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

Valid Values

`<schema>`

Where:

`<schema>`

is the name of a valid schema on the backend data store.

Default: No schema is specified. Therefore, all schemas are exposed.

Oracle Marketing Cloud (Eloqua) parameters

The following tables describe parameters available on the tabs of a Oracle Marketing Cloud Data Source dialog:

- General tab
- OData tab
- Mapping tab
- Advanced tab
General tab

Create Oracle Marketing Cloud Data Source

Table 55: General tab connection parameters for Oracle Marketing Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of this set of connection parameters.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
UserId, Password | The login credentials for your Oracle Marketing Cloud data store account.

**Note:** By default, the password is encrypted.

By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye 🕵️ icon. Click the icon again to conceal the password.

In addition to the user ID and password, the company identifier must be set.

Company | The company identifier that Oracle Marketing Cloud issues after registration. For example, if your company name is My Company LLC, Oracle Marketing Cloud might issue the company identifier as mycompany.

**Note:** If you do not know this value, ask the person who registered the Oracle Marketing Cloud account.

---

**OData tab**

The following table describes the controls on the **OData** tab. For information on using the **Configure Schema** editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see "Formulating queries" under Querying with OData.
Table 56: OData tab connection parameters for Oracle Marketing Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Access URI</td>
<td>Specifies the base URI for the OData feed to access the data source, for example, <a href="https://example.com:8443/api/odata4/">https://example.com:8443/api/odata4/</a>&lt;datasourcename&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Page Size    | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** $0 | $n  
$n$ is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0 |
| Refresh Result | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1 |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the $inlinecount parameter when it is set to allpages. These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
**Valid Values:**  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
**Default:** 2 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>When <strong>ON</strong> is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When <strong>OFF</strong> is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
</tbody>
</table>
Mapping tab

Create Oracle Marketing Cloud Data Source

The default values for advanced mapping fields are appropriate in many cases. However, if your organization wants to strip custom prefixes or enable uppercase identifiers, you might want to change map option settings. Understanding how the Hybrid Data Pipeline connectivity service creates and uses maps will help you choose the appropriate values.

Click the + next to Set Map Options to display these fields.

The following table describes the mapping options that apply to Oracle Marketing Cloud.

**Note:** Map creation is an expensive operation. In most cases, you will only want to re-create a map if you need to change mapping options.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map Name</td>
<td>Optional name of the map definition that Hybrid Data Pipeline uses to interpret the schema of the cloud data store. The Hybrid Data Pipeline service automatically creates a name for the map.</td>
</tr>
</tbody>
</table>
### Refresh Schema

The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects.

**Valid Values:**

- When set to **ON**, the connectivity service attempts to refresh the schema.
- When set to **OFF**, the connectivity service does not attempt to refresh the schema.

**Default**

**OFF**

**Notes**

- You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.
- Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.
- If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.

### Create Mapping

Determines whether the Oracle Marketing Cloud table mapping files are to be (re)created.

The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Not Exist</strong></td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the <strong>Map Name</strong> field, the name will be a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong>.</td>
</tr>
<tr>
<td><strong>Force New</strong></td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to <strong>Force New</strong> indefinitely.</td>
</tr>
<tr>
<td><strong>No</strong></td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>
The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks.

For example, the `Case` object is a standard object present in most Salesforce organizations but `CASE` is also an SQL keyword. Therefore, a table named `Case` cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:

- **Execution of the SQL query** `Select * from Case` will return the following:
  
  Error: [DataDirect][DDCloud JDBC Driver][Salesforce]Unexpected token: CASE in statement [select * from case]

- **Execution of the SQL query** `Select * from "Case"` will return the following:
  
  Error: [DataDirect][DDCloud JDBC Driver][Salesforce]Table not found in statement [select * from "Case"]

- **Execution of the SQL query, Select * from "CASE"** will complete successfully.

To avoid using quotes and uppercase for table or column names that match keywords and reserved words, you can instruct Hybrid Data Pipeline to add a suffix to such names. For example, if **Keyword Conflict Suffix** is set to TAB, the `Case` table will be mapped to a table named CASETAB. With such a suffix appended in the map, the following queries both work:

- `Select * from CASETAB`
- `Select * from casetab`

The default value is an empty string.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Box As Text</td>
<td>Specifies whether the check box values of the user-defined columns should be returned as a string or as boolean.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the check box value is returned as a boolean, which is described as BIT in the schema. Any values that cannot be matched to the current 'checkedValue' or 'uncheckedValue' are returned as NULL.</td>
</tr>
<tr>
<td></td>
<td>If set to 1, the stored literal value of the check box is returned as a string, which is described as WVARCHAR in the schema.</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
</tbody>
</table>
Advanced tab

Create Oracle Marketing Cloud Data Source

- **Web Service Fetch Size**: 1000
- **Web Service Retry Count**: 0
- **Web Service Timeout**: 120
- **Fail On Incomplete Data**: 0
- **Enable Bulk Load**: 1
- **Activity Bulk Page Size**: 50000
- **Bulk Page Size**: 5000
- **Bulk Timeout**: 18000
- **Bulk Top Threshold**: 1000
- **Read Only**: Off
- **Extended Options**: 
- **Metadata Exposed Schemas**: 

[Advanced tab settings interface]
Table 59: Advanced tab connection parameters for Oracle Marketing Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Fetch Size</td>
<td>Specifies the number of rows of data the Hybrid Data Pipeline connectivity service attempts to fetch for each call.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the Hybrid Data Pipeline connectivity service attempts to fetch up to a maximum of 10000 rows. This value typically provides the maximum throughput.</td>
</tr>
<tr>
<td></td>
<td>If set to x, the Hybrid Data Pipeline connectivity service attempts to fetch up to a maximum of the specified number of rows. Setting the value lower than 10000 can reduce the response time for returning the initial data. Consider using a smaller value for interactive applications only.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 1000</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. Insert, Update, and Delete requests are never retried. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 0.</td>
</tr>
<tr>
<td>Web Service Timeout</td>
<td>The time, in seconds, to wait before retrying a timed-out Select request. Valid only if the value of Web Service Retry Count is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.</td>
</tr>
<tr>
<td>Fail On Incomplete Data</td>
<td>Specifies how Hybrid Data Pipeline processes a query when Oracle Marketing Cloud returns no data for some columns.</td>
</tr>
<tr>
<td></td>
<td>For these columns, which together form incomplete data, the connectivity service can either return NULL values or throw an exception.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service returns NULL values for such columns.</td>
</tr>
<tr>
<td></td>
<td>If set to 1, if possible, the connectivity service tries to retrieve the complete data using the bulk load.</td>
</tr>
<tr>
<td></td>
<td>While using the bulk load, if the number of columns exceeds 100 and the interface is therefore unable to satisfy the requirements of the query, the connectivity service throws an exception.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> It is preferable that you enable bulk load (Enable Bulk Load), as this allows more options for retrieving the data.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Enable Bulk Load</td>
<td>Enables or disables bulk support for fetching data. If set to 1, bulk support is enabled. If set to 0, bulk support is disabled. Default: 1</td>
</tr>
<tr>
<td>Activity Bulk Page Size</td>
<td>The number of records to be fetched from Activity_XXX tables in a single request when using the bulk load. Valid Values: 2 to 50000 Default: 50000</td>
</tr>
<tr>
<td>Bulk Page Size</td>
<td>The number of records to be fetched from Oracle Marketing Cloud in a single request when using the bulk load. Valid Values: 2 to 50000 Default: 5000</td>
</tr>
<tr>
<td>Bulk Timeout</td>
<td>The timeout duration for a bulk call in seconds. Oracle Marketing Cloud automatically clears out the bulk staging area after this timeout, so if the query is large and the data takes more than this time to run, the query could be aborted midstream. This property only has an effect if the bulk load is enabled. Valid Values: 3600 to 120960 Default: 18000</td>
</tr>
<tr>
<td>Bulk Top Threshold</td>
<td>For a Select query that qualifies for the bulk operations and the TOP n clause is used: If the specified value is less than or equal to 1000, the standard mechanism would be used to process the query. If the specified value is greater than 1000, bulk load would be used to process the query. Valid Values: An integer greater than 0 Default: 1000</td>
</tr>
<tr>
<td>Read Only</td>
<td>Enables write operations to Oracle Marketing Cloud. If set to ON, the data source is read only. Write operations are not allowed. If set to OFF, write operations are permitted. Default: OFF</td>
</tr>
</tbody>
</table>

5 Generally, higher page sizes return results more quickly. However, Oracle Marketing Cloud imposes a 32 MB limit on response package size. If queries return large records, too many records within a single page will exceed that limit, causing the query to fail.

6 All of the objects returned within a page must be materialized as the page is retrieved, so sufficient Java heap space is necessary with large page sizes containing many small columns.
### Extended Options

Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1; UndocumentedOption1=value[; UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.

**Valid Values:** `string`

**Default:** empty string

### Metadata Exposed Schemas

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

```
<schema>
```

**Where:**

```
<schema>
```

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

---

**Oracle Sales Cloud parameters**

The following tables describe parameters available on the tabs of an Oracle® Sales Cloud™ Data Source dialog:

- **General tab**
- **OData tab**
- **Mapping tab**
- **Advanced tab**
Chapter 3: Using Hybrid Data Pipeline

General tab

Create Oracle Sales Cloud Data Source

<table>
<thead>
<tr>
<th>General</th>
<th>OData</th>
<th>Mapping</th>
<th>Advanced</th>
</tr>
</thead>
</table>

Data Source Name *

Description

User ID

Password

Oracle Sales Cloud Login URL *

* Required Fields

Cancel  TEST  Save
Table 60: General tab connection parameters for Oracle Sales Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials for your Oracle Sales Cloud data store account.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> By default, the password is encrypted.</td>
</tr>
<tr>
<td></td>
<td>The Hybrid Data Pipeline connectivity service uses this information to connect to the data store. The administrator of the data store must grant permission to a user with these credentials to access the data store and the target data.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data Source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline connectivity service.</td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Oracle Sales Cloud Login URL</td>
<td>The Host Name for the Oracle Sales Cloud site that the Hybrid Data Pipeline connectivity service will use to query the service; for example, mysite.custhelp.com.</td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the **Configure Schema** editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846 under Querying with OData.
### Create Oracle Sales Cloud Data Source

![Create Oracle Sales Cloud Data Source](image)

**Table 61: OData tab connection parameters for Oracle Sales Cloud**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
**OData Access URI** | Specifies the base URI for the OData feed to access your data source, for example, `https://hybridpipe.operations.com/api/odata/<DataSourceName>`. You can copy the URI and paste it into your application's OData configuration.

The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding `$metadata` to the service root URI.

**Schema Map** | Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.

See [Configuring data sources for OData connectivity and working with data source groups](#) on page 622 for more information.

**Data Source Caching** | Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

**Valid Values:**

- **Valid Values:**
  - When set to 1, session caching is enabled. This provides better performance for production.
  - When set to 0, session caching is disabled. Use this value when you are configuring the data source.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Page Size     | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages. **Valid Values:** $0$ | $n$  
|               | where $n$ is an integer from 1 to 10000.                                                                                                           |  
|               | When set to 0, the server default of 2000 is used.                                                                                               |  
|               | **Default:** 0                                                                             |  
| Refresh Result| Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change. **Valid Values:**  
|               | When set to 0, the OData service caches the first page of results.                                                                             |  
|               | When set to 1, the OData service re-executes the query.                                                                                         |  
|               | **Default:** 1                                                                             |  
| Inline Count Mode| Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging. The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large. **Valid Values:**  
|               | When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
|               | When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
|               | **Default:** 1                                                                             |  

*Chapter 3: Using Hybrid Data Pipeline*
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
</tbody>
</table>

| OData Read Only  | Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled. |
|                  | Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon. |
|                  | **Valid Values:**                                                                                                                                 |
|                  | ON | OFF                                                                                                                                 |
|                  | When ON is selected, OData access is restricted to read-only mode.                                                                           |
|                  | When OFF is selected, write operations can be performed on the OData service.                                                               |
|                  | **Default:** OFF                                                                                                                           |
Mapping tab

Create Oracle Sales Cloud Data Source

The default values for advanced mapping fields are appropriate in many cases. However, if your organization wants to strip custom prefixes or enable uppercase identifiers, you might want to change map option settings. Understanding how the Hybrid Data Pipeline connectivity service creates and uses maps will help you choose the appropriate values.

Click the + next to Set Map Options to display these fields.

The following table describes the mapping options that apply to Oracle Sales Cloud.

**Note:** Map creation is an expensive operation. In most cases, you will only want to re-create a map if you need to change mapping options.

Table 62: Mapping tab Connection Parameters for Oracle Sales Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map Name</td>
<td>Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map.</td>
</tr>
</tbody>
</table>
Specifies whether the Hybrid Data Pipeline connectivity service attempts to refresh the schema when an application first connects.

**Valid Values:**

- **ON** | **OFF**

If set to **OFF** and the `ResultSetMetaData.getTableName()` method is called, the connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The `getTableName()` method may return an empty string for each column in the result set.

If set to **ON** and the `ResultSetMetaData.getTableName()` method is called, the connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the `ResultSetMetaData.getSchemaName()` and `ResultSetMetaData.getCatalogName()` methods are called if the connectivity service can determine that information.

**Default:** **OFF**

Determines whether the Oracle Sales Cloud table mapping files are to be (re)created.

The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

**Table 63: Valid values for Create Map field**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Not Exist</strong></td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the <strong>Map Name</strong> field, the name will be a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong>.</td>
</tr>
<tr>
<td><strong>Force New</strong></td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to <strong>Force New</strong> indefinitely.</td>
</tr>
<tr>
<td><strong>No</strong></td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>API Version</strong></td>
<td>Identifies the version of Oracle Sales Cloud used in your environment.</td>
</tr>
<tr>
<td></td>
<td><strong>By default, API Version is set to latest. When set to latest, the connectivity service assumes the latest version of Oracle Sales Cloud is being used.</strong></td>
</tr>
<tr>
<td></td>
<td><strong>API Version</strong> can also be set to a specific Oracle Sales Cloud API version, for example, 11.1.11.</td>
</tr>
<tr>
<td><strong>API Endpoints</strong></td>
<td>Specifies modules for Oracle Sales Cloud instances. The Hybrid Data Pipeline connectivity service retrieves resources from the specified endpoints. Modules must be separated by a comma.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> salesApi, crmCommonApi</td>
</tr>
<tr>
<td><strong>Varchar Threshold</strong></td>
<td>Specifies the threshold at which columns of the data type SQL_VARCHAR are described as SQL_LONGVARCHAR. If the size of the SQL_VARCHAR column exceeds the value specified, the column is described as SQL_LONGVARCHAR when calling SQLDescribeCol and SQLColumns. This option allows you to fetch columns that would otherwise exceed the upper limit of the SQL_VARCHAR type for some third-party applications.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 4000</td>
</tr>
</tbody>
</table>
Table 64: Advanced tab connection parameters for Oracle Sales Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Fetch Size</td>
<td>Specifies the number of rows of data the Hybrid Data Pipeline connectivity service attempts to fetch for each Web service call.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service attempts to fetch up to a maximum of 100 rows. This value typically provides the maximum throughput.</td>
</tr>
<tr>
<td></td>
<td>If set to x, the connectivity service attempts to fetch up to a maximum of the specified number of rows. Setting the value lower than 100 can reduce the response time for returning the initial data. Consider using a smaller value for interactive applications only.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 100</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. Insert, Update, and Delete requests are never retried. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 0.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Web Service Timeout</strong></td>
<td>The time, in seconds, to wait before retrying a timed-out Select request. Valid only if the value of Web Service Retry Count is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.</td>
</tr>
<tr>
<td><strong>Max Pooled Statements</strong></td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application. <strong>Default:</strong> 0</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Extended Options       | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
  Database=Server1; UndocumentedOption1=value[; UndocumentedOption2=value;]  
  If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  
  **Valid Values:** string  
  **Default:** empty string  
  **Note:**  
  If you are using a proxy server to connect to your Sales Cloud instance, then you have to set these options:  
  proxyHost = hostname of the proxy server; proxyPort = port number of the proxy server  
  If Authentication is enabled, then you have to include the following:  
  proxyuser=<value>; proxypassword=<value> |
| Metadata Exposed Schemas| Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.  
  **Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.  
  **Valid Values**  
  <schema>  
  Where:  
  <schema>  
  is the name of a valid schema on the backend data store.  
  **Default:** No schema is specified. Therefore, all schemas are exposed. |

How to create a data source in the Web UI on page 225
Oracle Service Cloud parameters

The following tables describe parameters available on the tabs of an Oracle® Service Cloud™ Data Source dialog:

- General tab
- OData tab
- Mapping tab
- Advanced tab

General tab

Create Oracle Service Cloud Data Source
### Table 65: General tab connection parameters for Oracle Service Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials for your Oracle Service Cloud data store account.</td>
</tr>
<tr>
<td><strong>Note:</strong> By default, the password is encrypted.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The Hybrid Data Pipeline connectivity service uses this information to connect to the data store. The administrator of the data store must grant permission to a user with these credentials to access the data store and the target data.</td>
</tr>
<tr>
<td><strong>Note:</strong> You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data Source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline connectivity service.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Oracle Service Cloud Login URL</td>
<td>The Host Name for the Oracle Service Cloud site that Hybrid Data Pipeline will use to query the service; for example mysite.custhelp.com, mysite.custhelp.com.</td>
</tr>
<tr>
<td>Interface</td>
<td>The name of the Oracle Service Cloud interface to which you want to connect.</td>
</tr>
</tbody>
</table>

### OData tab

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 66: OData tab connection parameters for Oracle Service Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Page Size</td>
<td>Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages. <strong>Valid Values:</strong> 0</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Refresh Result      | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1 |
| Inline Count Mode   | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
**Valid Values:**  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
**Default:** 1 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Top Mode        | Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.  
**Valid Values:**  
Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.  
Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.  
**Default:** 0   |
| OData Read Only | Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.  
Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.  
**Valid Values:**  
ON | OFF  
When ON is selected, OData access is restricted to read-only mode.  
When OFF is selected, write operations can be performed on the OData service.  
**Default:** OFF |

**Description**

Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**

Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.  
Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.  
**Default:** 0

**OData Read Only**

Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.  
Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.  
**Valid Values:**  
ON | OFF  
When ON is selected, OData access is restricted to read-only mode.  
When OFF is selected, write operations can be performed on the OData service.  
**Default:** OFF
The default values for advanced mapping fields are appropriate in many cases. However, if your organization wants to strip custom prefixes or enable uppercase identifiers, you might want to change map option settings. Understanding how Hybrid Data Pipeline creates and uses maps will help you choose the appropriate values.

Click the + next to **Set Map Options** to display these fields.

The following table describes the mapping options that apply to Oracle Service Cloud.

---

**Note:** Map creation is an expensive operation. In most cases, you will only want to re-create a map if you need to change mapping options.
Table 67: Mapping tab Connection Parameters for Oracle Service Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit Columns</td>
<td>The audit columns added by Hybrid Data Pipeline are:</td>
</tr>
<tr>
<td></td>
<td>• CreatedByAccountID</td>
</tr>
<tr>
<td></td>
<td>• CreatedTime</td>
</tr>
<tr>
<td></td>
<td>• UpdatedByAccountID</td>
</tr>
<tr>
<td></td>
<td>• UpdatedTime</td>
</tr>
</tbody>
</table>

The following table describes the valid values for the **Audit Columns** parameter.

Table 68: Valid values for Audit Columns

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Hybrid Data Pipeline includes all of the audit columns in its table definitions.</td>
</tr>
<tr>
<td>standard</td>
<td>Hybrid Data Pipeline adds only the audit columns in its table definitions.</td>
</tr>
<tr>
<td>custom</td>
<td>Hybrid Data Pipeline adds audit columns only for custom objects in its table definitions.</td>
</tr>
<tr>
<td>None</td>
<td>Hybrid Data Pipeline does not add the audit columns in its table definitions.</td>
</tr>
</tbody>
</table>

The default value for **Audit Columns** is **All**.

In a typical Oracle Service Cloud instance, not all users are granted access to the Audit columns. If **Audit Columns** is set to a value other than **None** and if Hybrid Data Pipeline cannot include the columns requested, the connection fails and an exception is thrown.
Determines whether the Oracle Service Cloud table mapping files are to be (re)created. The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

Table 69: Valid values for Create Map field

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the Map Name field, the name will be a combination of the User Name and DataSource ID.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the User Name and DataSource ID to name the map. Map creation is expensive, so you will likely not want to leave this option set to Force New indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>

Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map.

Defines whether Hybrid Data Pipeline maps the integration name of standard columns that appear in each Oracle Service Cloud object to a new name. By default, Hybrid Data Pipeline maps the id column to ROWID, and maps the remaining standard columns to a new name prefixed with SYS_.

**Valid Values:**

1 | 0

When set to 1, Hybrid Data Pipeline prefixes the names of standard columns of Oracle Service Cloud objects with SYS_ or ROWID.

When set to 0, Hybrid Data Pipeline does not map the names of standard columns of Oracle Service Cloud objects to new names.

**Default:** 0
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NamedID Behavior</strong></td>
<td>Controls whether the Name attribute of NamedID fields are exposed in the relational model.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>1</td>
</tr>
<tr>
<td>When set to 1, the Id and Name attributes of the NamedID fields are exposed in the relational model. This means that they will be included in the results for the queries. However, including these columns in queries can cause Oracle Service Cloud to return a “poor performing query” error if the table has a large number of rows. When set to 2, only the Id attribute of the NamedID fields is exposed in the relational model. This setting may improve performance of queries that use NamedID fields. <strong>Default:</strong> 1</td>
<td></td>
</tr>
<tr>
<td>Refresh Schema</td>
<td>Specifies whether the Hybrid Data Pipeline connectivity service attempts to refresh the schema when an application first connects.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>ON</td>
</tr>
<tr>
<td>If set to OFF and the ResultSetMetaData.getName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getName() method may return an empty string for each column in the result set. If set to ON and the ResultSetMetaData.getName() method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The Hybrid Data Pipeline connectivity service returns schema name and catalog name information when the ResultSetMetaData.getName() and ResultSetMetaData.getCatalogName() methods are called if the Hybrid Data Pipeline connectivity service can determine that information. <strong>Default:</strong> OFF</td>
<td></td>
</tr>
</tbody>
</table>
Advanced tab

Create Oracle Service Cloud Data Source

- **Web Service Call Limit**: 
  - 1
- **Web Service Fetch Size**: 
  - 0
- **Web Service Retry Count**: 
  - 0
- **Web Service Timeout**: 
  - 120
- **Max Pooled Statements**: 
  - 0
- **Oracle Service Cloud Database**: 
  - report
- **Enable Paging With Order By ID**: 
  - ON
- **Processing Options**: 
  - 0
- **Initialization String**: 
  - 
- **Read Only**: 
  - ON
- **Extended Options**: 
  - 
- **Metadata Exposed Schemas**: 
  -
Table 70: Advanced tab connection parameters for Oracle Service Cloud

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Paging With Order By ID</td>
<td>Specifies whether the Hybrid Data Pipeline connectivity service can inject the Order By clause in the Select query for each call. Enabling this connection parameter provides a stable paging mechanism for retrieving result sets that are larger than the maximum number of rows for the site. Note: If your application does not retrieve large result sets, consider disabling this feature, because adding the Order By clause can have a negative performance impact on queries. If set to ON, the Hybrid Data Pipeline connectivity service can inject the Order By clause in the Select query. Default: ON</td>
</tr>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example: Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;] If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence. Valid Values: string Default: empty string Note: If you are using a proxy server to connect to your service cloud instance, then you have to set these options: proxyHost = hostname of the proxy server; proxyPort = port number of the proxy server If Authentication is enabled, then you have to include the following: proxyuser=&lt;value&gt;; proxypassword=&lt;value&gt;</td>
</tr>
<tr>
<td>Initialization String</td>
<td>A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed. Syntax: SQLcommand[; SQLcommand]... where: SQLcommand is a SQL command. Multiple commands must be separated by semicolons. Default: an empty string.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Max Pooled Statements         | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.  
**Default:** 0                                                                 |
| Oracle Service Cloud Database | Determines against which database queries should be resolved. Oracle Service Cloud can satisfy queries against the production (operational) or the reporting database that backs the service.  
**Valid Values:**  
report | operational  
If set to **report**, the Hybrid Data Pipeline connectivity service prepends a "USE REPORT; " statement to the base ROQL command. This results in the reporting database being used for subsequent queries.  
If set to **operational**, the Hybrid Data Pipeline connectivity service prepends a "USE OPERATIONAL; " statement to the base ROQL command. This results in the production database being used for subsequent queries.  
If not specified, the Hybrid Data Pipeline connectivity service sends the base ROQL command directly. This results in the default database behavior.  
**Default:** report                                                                 |
| Processing Options            | Determines whether external events and business rules are run on the server side when performing a Create, Destroy, Get, or Update operation.  
**Valid Values:**  
0 | 1 | 2 | 3  
If set to 0, external events and business rules run after a Create, Destroy, Get, or Update operation completes.  
If set to 1, external events do not run after a Create, Destroy, Get, or Update operation completes.  
If set to 2, business rules do not run after a Create, Destroy, Get, or Update operation completes.  
If set to 3, external events and business rules do not run after a Create, Destroy, Get, or Update operation completes.  
**Default:** 0                                                                 |
| Read Only                     | Enables write operations to Oracle Service Cloud.  
If set to **ON**, the data source is read only. Write operations are not allowed.  
If set to **OFF**), write operations are permitted if Oracle Service Cloud Database is set to operational. Write operations are not supported if Oracle Service Cloud Database is set to report.  
**Default:** ON                                                                 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Call Limit</td>
<td>The maximum number of Web service calls allowed to the cloud data store for a single SQL statement or metadata query.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> $-1 \mid 0 \mid x$</td>
</tr>
<tr>
<td></td>
<td>where $x$ is a positive integer that defines the maximum number of Web service calls that the connectivity service can make when executing any single SQL statement or metadata query.</td>
</tr>
<tr>
<td></td>
<td>If set to $-1$, the connectivity service uses the default value that is configured in the service when connected to a site whose version is August 2014 or later. When connected to sites whose version is prior to August 2014, the connectivity service sets the maximum number of calls to 100.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service uses the maximum number of calls allowed by the service when connected to a site whose version is August 2014 or later. When connected to sites whose version is prior to August 2014, there is no limit.</td>
</tr>
<tr>
<td></td>
<td>If set to $x$, the connectivity service uses this value to set the maximum number of Web service calls that can be made when executing a SQL statement or metadata query. If you specify a value that is greater than the maximum number of calls allowed when connected to a site whose version is August 2014 or later, the connectivity service returns a warning and uses the maximum value instead.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> $-1$.</td>
</tr>
<tr>
<td>Web Service Fetch Size</td>
<td>Specifies the number of rows of data the Hybrid Data Pipeline connectivity service attempts to fetch for each web service call.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> $0 \mid x$</td>
</tr>
<tr>
<td></td>
<td>where $x$ is a positive integer that defines the maximum number of Web service calls that the connectivity service can make when executing any single SQL statement or metadata query. For servers prior to version 14.08, the maximum is 10,000 rows. For versions 14.08 and higher, the maximum is server dependent.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service uses the maximum page size for the Oracle Service Cloud database to which it is connecting (Operational or Report) for sites whose version is 14.08 or higher. When connecting to sites whose version is prior to 14.08, the connectivity service attempts to fetch up to a maximum of 10,000 rows. This value typically provides the maximum throughput.</td>
</tr>
<tr>
<td></td>
<td>If set to $x$, the connectivity service attempts to fetch up to a maximum of the specified number of rows. Setting the value lower than 10,000 can reduce the response time for returning the initial data. Consider using a smaller value for interactive applications only. If you specify a value greater than the server allows, the connectivity service returns a warning and uses the maximum value permitted.</td>
</tr>
<tr>
<td></td>
<td>The default is 0.</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. Insert, Update, and Delete requests are never retried. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 0.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Web Service Timeout | The time, in seconds, to wait before retrying a timed-out Select request. Valid only if the value of Web Service Retry Count is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.

Metadata Exposed Schemas | Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

<schema>

Where:

<schema>

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

---

**See the steps for:**

How to create a data source in the Web UI on page 225

---

### PostgreSQL parameters

The following tables describe parameters available on the tabs of a PostgreSQL On-Premise Data Source dialog:

- **General tab**
- **Security tab**
- **OData tab**
- **Advanced tab**
General tab

Create PostgreSQL Data Source
Table 71: General tab connection parameters for PostgreSQL

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Password</td>
<td>A case-sensitive password that is used to connect to your PostgreSQL database. A password is required if user ID/password authentication is enabled on your database. Contact your system administrator to obtain your password. Note: By default, the password is encrypted.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The port number of the PostgreSQL server.</td>
</tr>
</tbody>
</table>
| Server Name    | Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, PostgreServer or 122.23.15.12. Valid Values:  

   server_name | IP_address

where:

   server_name

   is the name of the server to which you want to connect.

   IP_address

   is the IP address of the server to which you want to connect.

The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two. |
The login credentials for your PostgreSQL server.

Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the server must grant permission to a user with these credentials to access the data store and the target data.

By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.

**Note:** You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline account.

### Security tab

#### Table 72: Security tab connection parameters for PostgreSQL On-Premise

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crypto Protocol Version</td>
<td>Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.</td>
</tr>
</tbody>
</table>
|                            | **Valid Values:**
|                            | cryptographic_protocol [[, cryptographic_protocol]...]
<p>|                            | where:                                                                                                                                                                                                     |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| cryptographic_protocol | is one of the following cryptographic protocols:  
                         TLSv1 | TLSv1.1 | TLSv1.2  
                         The client must send the highest version that it supports in the client hello. |

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.

**Example**

Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

TLSv1.1, TLSv1.2

**Default:** TLSv1, TLSv1.1, TLSv1.2

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Encryption Method | Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server.  
                          **Valid Values:**  
                          noEncryption | SSL | requestSSL  
                          If set to noEncryption, data is not encrypted or decrypted. 
                          If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception. 
                          If set to requestSSL, the login request and data is encrypted using SSL. If the database server does not support SSL, the connectivity service establishes an unencrypted connection. |

**Note:**

- When SSL is enabled, the following properties also apply:

  - Host Name In Certificate
  - ValidateServerCertificate
  - Crypto Protocol Version

**Default:** noEncryption
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Host Name In Certificate | Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.  
Valid Values:  
  
  host_name | #SERVERNAME#  
  
  where **host_name** is a valid host name.  
  
  If **host_name** is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.  
  
  If **#SERVERNAME#** is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.  
  
  **Default**: Empty string                                                                                                                                          |
| Validate Server Certificate |                                                                                                                                                                                                                                                                                                                                             |
Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the Hybrid Data Pipeline connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

**Valid Values:**

ON | OFF

If set to ON, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate parameter is specified, the Hybrid Data Pipeline connectivity service also validates the certificate using a host name. The Host Name In Certificate parameter is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the connectivity service is connecting to is the server that was requested.

If set to OFF, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any truststore information that is specified by the Java system properties. Truststore information is specified using Java system properties.

**Default:** ON

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the Hybrid Data Pipeline connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment. <strong>Valid Values:</strong> ON</td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 73: OData tab connection parameters for PostgreSQL

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.&lt;br&gt;See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.&lt;br&gt;Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.&lt;br&gt;When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.&lt;br&gt;&lt;strong&gt;Valid Values&lt;/strong&gt;:&lt;br&gt;When set to 1, session caching is enabled. This provides better performance for production.&lt;br&gt;When set to 0, session caching is disabled. Use this value when you are configuring the data source.&lt;br&gt;&lt;strong&gt;Default&lt;/strong&gt;: 1</td>
</tr>
<tr>
<td>Page Size</td>
<td>Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.&lt;br&gt;&lt;strong&gt;Valid Values&lt;/strong&gt;: 0</td>
</tr>
</tbody>
</table>
### Refresh Result

Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.

**Valid Values:**

- When set to 0, the OData service caches the first page of results.
- When set to 1, the OData service re-executes the query.

**Default:** 1

### Inline Count Mode

Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.

**Valid Values:**

- When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.
- When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

**Default:** 1

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Refresh Result    | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change. **Valid Values:**
- When set to 0, the OData service caches the first page of results.
- When set to 1, the OData service re-executes the query.
**Default:** 1 |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging. The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large. **Valid Values:**
- When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.
- When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large. **Default:** 1 |
## Top Mode

Indicates how requests typically use `$top` and `$skip` for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**

Set to 0 when the application generally uses `$top` to limit the size of the result and rarely attempts to get additional entities by combining `$top` and `$skip`.

Set to 1 when the application uses `$top` as part of client-driven paging and generally combines `$top` and `$skip` to page through the result.

**Default:** 0

## OData Read Only

Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.

Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.

**Valid Values:**

- **ON**
- **OFF**

When **ON** is selected, OData access is restricted to read-only mode.

When **OFF** is selected, write operations can be performed on the OData service.

**Default:** OFF
Advanced tab

Create PostgreSQL Data Source

- **Alternate Servers**: 
- **Load Balancing**: 
- **Catalog Options**:  
- **Extended Options**: 
- **Initialization String**: 
- **Login Timeout**: 30
- **Max Pooled Statements**: 0
- **Query Timeout**: 0
- **Result Set Meta Data Options**: 0
- **Metadata Exposed Schemas**: 

**Required Fields**

**Cancel**  **TEST**  **Save**
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alternate Servers</td>
<td>Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>(servername1:[port1],servername2:[port2])...</td>
</tr>
<tr>
<td></td>
<td>The server name (servername1, servername2, and so on) is required for each alternate server entry. Port number (port1, port2, and so on) is optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number is used.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> None</td>
</tr>
<tr>
<td>Catalog Options</td>
<td>Determines which type of metadata information is included in result sets when an application calls DatabaseMetaData methods. To include multiple types of metadata information, add the sum of the values that you want to include. In this case, specify 6 to query database catalogs for column information and to emulate getColumns() calls.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> 2</td>
</tr>
<tr>
<td></td>
<td>If set to 2, the Hybrid Data Pipeline connectivity service queries database catalogs for column information.</td>
</tr>
<tr>
<td></td>
<td>If set to 4, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate getColumns() calls using the ResultSetMetaData object instead of querying database catalogs for column information. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumns(). The argument to getColumns() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the connectivity service reverts to the default behavior for getColumns() calls.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 2</td>
</tr>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:</td>
</tr>
<tr>
<td></td>
<td>Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]</td>
</tr>
<tr>
<td></td>
<td>If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> string</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> empty string</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| Initialization String | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  

**Syntax:**  
command[[; command]...]  

Where:  
command  
is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:  
InitializationString=(REFRESH SCHEMA SFORCE)  
The default is an empty string. |
| Load Balancing    | Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property.  

**Valid Values:** ON | OFF  
If set to ON, the connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established.  
If set to OFF, the connectivity service does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).  

**Default:** OFF  

**Notes**  
- The Alternate Servers connection parameter specifies one or multiple alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers parameter. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **LoginTimeout**             | **Valid Values:**
|                              | \[0 \ | \ x\] where \(x\) is a positive integer that represents a number of seconds. If set to 0, the connectivity service does not time out a connection request. If set to \(x\), the connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception. **Default:** 30 |
| **Max Pooled Statements**   | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application. The default value is 0. |
| **Metadata Exposed Schemas** | Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema. **Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data. **Valid Values**
|                              | \(<schema>\) Where:
<p>|                              | (&lt;schema&gt;) is the name of a valid schema on the backend data store. <strong>Default:</strong> No schema is specified. Therefore, all schemas are exposed. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Query Timeout | Sets the default query timeout (in seconds) for all statements created by a connection.  
Valid Values:  

-1 | 0 | x  

If set to -1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the Statement.setQueryTimeout() method.  
If set to 0, the default query timeout is infinite (the query does not time out).  
If set to x, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection option, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.  
Default: 0
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResultSet Meta</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.</td>
</tr>
<tr>
<td>Data Options</td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>If set to 0 and the ResultSetMetaData.getTableName() method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The getTableName() method may return an empty string for each column in the result set.</td>
</tr>
<tr>
<td></td>
<td>If set to 1 and the ResultSetMetaData.getTableName() method is called, the connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the ResultSetMetaData.getSchemaName() and ResultSetMetaData.getCatalogName() methods are called if the connectivity service can determine that information.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>Transaction Error</td>
<td>Determines how the driver handles errors that occur within a transaction. When an error occurs in a transaction, the PostgreSQL server does not allow any operations on the connection except for rolling back the transaction.</td>
</tr>
<tr>
<td>Behavior</td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>none</td>
</tr>
<tr>
<td></td>
<td>If set to none, the connectivity service does not roll back the transaction when an error occurs. The application must handle the error and roll back the transaction. Any operation on the statement other than a rollback results in an error.</td>
</tr>
<tr>
<td></td>
<td>If set to RollbackTransaction, the connectivity service rolls back the transaction when an error occurs. In addition to the original error message, the connectivity service posts an error message indicating that the transaction has been rolled back.</td>
</tr>
<tr>
<td></td>
<td>If set to RollbackSavepoint, the connectivity service rolls back the transaction to the last savepoint when an error is detected. In manual commit mode, the connectivity service automatically sets a savepoint after each statement issued. This value makes transaction behavior resemble that of most other database system types, but uses more resources on the database server and may incur a slight performance penalty.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> RollbackTransaction</td>
</tr>
</tbody>
</table>

See the steps for:

*How to create a data source in the Web UI* on page 225

**Progress OpenEdge parameters**

The following tables describe parameters available on the tabs of a Progress® OpenEdge® Data Source setup dialog:

- General tab
Creating data sources with the Web UI

- Security tab
- OData tab
- Advanced tab

**General tab**

Create OpenEdge Data Source

[Diagram of the Create OpenEdge Data Source form with required fields highlighted]

- Data Source Name
- Description
- User ID
- Password
- Server Name
- Port Number
- Database
- Connector ID
Table 75: General tab connection parameters for Progress OpenEdge

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Password</td>
<td>A case-sensitive password that is used to connect to your Progress OpenEdge database. A password is required if user ID/password authentication is enabled on your database. Contact your system administrator to obtain your password. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password. <strong>Note:</strong> By default, the password is encrypted.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The port number configured in OpenEdge interface to serve the specified database.</td>
</tr>
<tr>
<td>Server Name</td>
<td>The name of the server machine on which the OpenEdge database to connect to is running. The value is the name of the server as it is known on the On-Premise network, for example, myopenedge.</td>
</tr>
<tr>
<td>User Id</td>
<td>The login credentials for your Progress OpenEdge server. Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the server must grant permission to a user with these credentials to access the data store and the target data.</td>
</tr>
</tbody>
</table>
# Security tab

## Create OpenEdge Data Source

![Security tab interface](image)

### Table 76: Security tab connection parameters for Progress OpenEdge

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crypto Protocol Version</td>
<td>Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.</td>
</tr>
</tbody>
</table>

#### Valid Values:

```
cryptographic_protocol [[, cryptographic_protocol]...]
```

where:

- `cryptographic_protocol` is one of the following cryptographic protocols:
  - TLSv1 | TLSv1.1 | TLSv1.2
  - The client must send the highest version that it supports in the client hello.

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.

#### Example

Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

```
TLSv1.1, TLSv1.2
```

**Default:** TLSv1, TLSv1.1, TLSv1.2
### Encryption Method

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the driver and the on-premise database server.</td>
</tr>
<tr>
<td><strong>Valid Values</strong></td>
<td><strong>noEncryption</strong></td>
</tr>
<tr>
<td></td>
<td>If set to <strong>noEncryption</strong>, data is not encrypted or decrypted. If set to <strong>SSL</strong>, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the vwz1474495743590 connectivity service throws an exception.</td>
</tr>
</tbody>
</table>

**Note:**

- Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the Login Timeout parameter to avoid problems when connecting to a server that does not support SSL.
- When SSL is enabled, the Host Name In Certificate and Validate Server Certificate parameters also apply.

The default value is **noEncryption**.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Host Name In Certificate | Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.  

**Valid Values:**  

`host_name | #SERVERNAME#`

where `host_name` is a valid host name.

If `host_name` is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.

If `#SERVERNAME#` is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.

**Default:** Empty string

| Validate Server Certificate | Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (EncryptionMethod=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.  

**Valid Values:**  

`ON | OFF`

If set to `ON`, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the Host Name In Certificate parameter is specified, the driver also validates the certificate using a host name. The HostNameInCertificate property is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to `OFF`, the connectivity service does not validate the certificate that is sent by the database server. The Hybrid Data Pipeline connectivity service ignores any truststore information that is specified by the TrustStore and TrustStorePassword properties or Java system properties. Truststore information is specified using the Java system properties.

**Default:** ON |
OData tab

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Page Size        | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
  **Valid Values:** $0 | $n  
  where $n is an integer from 1 to 10000.  
  When set to 0, the server default of 2000 is used.  
  **Default:** 0 |
| Refresh Result   | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
  **Valid Values:**  
  When set to 0, the OData service caches the first page of results.  
  When set to 1, the OData service re-executes the query.  
  **Default:** 1 |
| Inline Count Mode| Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinelcount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
  The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
  **Valid Values:**  
  When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
  When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
  **Default:** 1 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>When <strong>ON</strong> is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When <strong>OFF</strong> is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
</tbody>
</table>
**Advanced tab**

**Create OpenEdge Data Source**

Table 78: Advanced tab connection parameters for Progress OpenEdge

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Alternate Servers   | Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property. **Valid Values:**  
  
  \((servername1[:port1][,servername2[:port2]][...)\)  
  
  The server name (servername1, servername2, and so on) is required for each alternate server entry. Port number (port1, port2, and so on) is optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number is used. **Default:** None
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Catalog Options  | Determines which type of metadata information is included in result sets when a JDBC application calls DatabaseMetaData methods. To include multiple types of metadata information, add the sum of the values that you want to include. In this case, specify 6 to include synonyms and to emulate getColumns() calls.  

**Valid Values:**

0 | 2 | 4

If set to 0, result sets do not contain synonyms.

If set to 2, result sets contain synonyms that are returned from the following DatabaseMetaData methods: getColumns(), getExportedKeys(), getFunctionColumns(), getFunctions(), getImportedKeys(), getIndexInfo(), getPrimaryKeys(), getProcedureColumns(), and getProcedures().

If set to 4, a hint is provided to the driver to emulate getColumns() calls using the ResultSetMetaData object instead of querying database catalogs for column information. Result sets contain synonyms. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumns(). The argument to getColumns() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the driver reverts to the default behavior for getColumns() calls.

The default is 2. |
| Default Schema   | The name of the schema used when identifiers are not qualified in a SQL query. For example, suppose Default Schema is set to White. Subsequent SQL statements with unqualified table references use the owner name White. In this example, `SELECT * FROM Customer` returns all rows in the 'White.Customer' table. The username establishing the original session is still the current user.  

**Syntax:**

```
string_literal
```

Where:

```
string_literal
```

specifies the name for the default owner as a string literal, enclosed in single or double quotes.

When the field is left blank, the data store uses the default schema for the user.

The default is an empty string.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Extended Options           | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
  Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]  
  If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  
  **Valid Values:** string  
  **Default:** empty string                                                                                                                                                                                                                                                                                                                                                                                       |
| Initialization String      | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
  **Syntax:** command[; command]...  
  **Where:** command  
  is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:  
  InitializationString=(REFRESH SCHEMA SFORCE)  
  The default is an empty string.                                                                                                                                                                                                                                                                                                                                                                               |
| Load Balancing             | Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property.  
  **Valid Values:** ON | OFF  
  If set to **ON**, the connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established.  
  If set to **OFF**, the connectivity service does not use client load balancing and connects to each servers based on their sequential order (primary server first, then, alternate servers in the order they are specified).  
  **Default:** OFF  
  **Notes**  

The Alternate Servers connection parameter specifies one or multiple alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers parameter.

**LoginTimeout**

The amount of time, in seconds, that the Hybrid Data Pipeline connectivity service waits for a connection to be established before timing out the connection request.

**Valid Values:**

0 | x

where x is a positive integer that represents a number of seconds.

If set to 0, the driver does not time out a connection request.

If set to x, the Hybrid Data Pipeline connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.

The default is 30.

**Max Pooled Statements**

The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.

The default value is 0.

**Metadata Exposed Schemas**

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

⚠️ **Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

<schema>

Where:

<schema>

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

See the steps for: How to create a data source in the Web UI on page 225
Progress Rollbase parameters

Creating a **Data Source** defines how to connect to your cloud **Data Store**. See [How to create a data source in the Web UI](#) on page 225.

The Progress® Rollbase® **On-Premise Data Source** dialog provides the connection parameters described in the following tables to connect to Rollbase data:

- General tab
- OData tab
- Mapping tab
- Advanced tab

### General tab

![Create Rollbase Data Source](image-url)
Table 79: General tab connection parameters for Progress Rollbase

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of this set of connection parameters.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>Login credentials for a Rollbase Private Cloud account with sufficient permissions to access the data of interest. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Host Name</td>
<td>The name of the host on which Rollbase is installed. In a multi-server environment, the host on which you installed the Master server. You can confirm the hostname by navigating to Setup &gt; Application Setup &gt; SOAP API &gt; URI. The host name is the part of the URL following http:// and preceding the port number. For example, in the following URL, mercury is the host name: <a href="http://mercury:8080/webapi/services/rpcrouter">http://mercury:8080/webapi/services/rpcrouter</a>.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The port number to access Rollbase Private Cloud. The default is 443, which is the port used for SSL.</td>
</tr>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 80: OData tab connection parameters for Rollbase

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
</tbody>
</table>
| Data Source Caching| Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.  
**Valid Values:**  
When set to 1, session caching is enabled. This provides better performance for production.  
When set to 0, session caching is disabled. Use this value when you are configuring the data source.  
**Default:** 1                                                                                                                                                                                                                                                                                                                                 |
| Page Size          | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** 0 | n  
where n is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0  

### Field: Refresh Result

Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.

**Valid Values:**

- When set to 0, the OData service caches the first page of results.
- When set to 1, the OData service re-executes the query.

**Default:** 1

### Field: Inline Count Mode

Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.

**Valid Values:**

- When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.
- When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

**Default:** 1
### Top Mode

Indicates how requests typically use `$top` and `$skip` for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**

- Set to 0 when the application generally uses `$top` to limit the size of the result and rarely attempts to get additional entities by combining `$top` and `$skip`.
- Set to 1 when the application uses `$top` as part of client-driven paging and generally combines `$top` and `$skip` to page through the result.

**Default:** 0

### OData Read Only

Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.

Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.

**Valid Values:**

- **ON** | **OFF**

  - When **ON** is selected, OData access is restricted to read-only mode.
  - When **OFF** is selected, write operations can be performed on the OData service.

**Default:** OFF

### Mapping tab

You can set Map Options, which are values that provide the information required to create a connection to Progress Rollbase. Click the + next to **Set Map Options** to display these fields.
Chapter 3: Using Hybrid Data Pipeline

Create Rollbase Data Source

General | OData | Mapping | Advanced

Map Name

Refresh Schema

Create Mapping

Set Map Options

Map System Column Names

Uppercase Identifiers

Use Integration Names

Required Fields

Cancel | TEST | Save
Table 81: Mapping tab connection parameters for Rollbase

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Mapping</td>
<td>Determines whether the Rollbase table mapping files are to be (re)created. The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.</td>
</tr>
</tbody>
</table>

Note: You must force creation of a new map when there is a change in the mapping options for the data source, or when the User Name / User ID connecting to the data source has changed. The mapping is tied to the user account that initially connects through the driver when the data source is created. If the user account is changed, then the map must be recreated. Simply change the value of the **Create Map** option to force creation of a new map.

Table 82: Valid values for Create Map field

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the <strong>Map Name</strong> field, the name will be a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong>.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the <strong>User Name</strong> and <strong>Data Source ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to Force New indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>

Map Name

Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map.

If you want to name the map yourself, enter a unique name.
## Map System Column Names

The `mapSystemColumnNames` parameter defines whether Hybrid Data Pipeline maps the integration name of standard columns that appear in each Rollbase object to a new name. By default, Hybrid Data Pipeline maps the id column to ROWID, and maps the remaining standard columns to a new name prefixed with SYS_.

Valid values for `mapSystemColumnNames` are:

- **1**
- **0**

When set to 1, Hybrid Data Pipeline prefixes the names of standard columns of Rollbase objects with SYS_ or ROWID.

When set to 0, Hybrid Data Pipeline does not map the names of standard columns of Rollbase objects to new names.

**Default:** 1

## Refresh Schema

Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.

**Valid Values:**

- **0**
- **1**

When set to 0, the OData service caches the first page of results.

When set to 1, the OData service re-executes the query.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uppercase Identifiers</td>
<td>Defines how Hybrid Data Pipeline maps identifiers. By default, all unquoted identifier names are mapped to uppercase. Identifiers are object names. Classes, methods, variables, interfaces, and database objects, such as tables, views, columns, indexes, triggers, procedures, constraints, and rules, can have identifiers.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>When set to <strong>ON</strong>, the connectivity service maps all identifier names to uppercase.</td>
</tr>
<tr>
<td></td>
<td>When set to <strong>OFF</strong>, Hybrid Data Pipeline maps identifiers to the mixed case name of the object being mapped. If mixed case identifiers are used, those identifiers must be quoted in SQL statements, and the case of the identifier, must exactly match the case of the identifier name.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> When object names are passed as arguments to catalog functions, the case of the value must match the case of the name in the database. If an unquoted identifier name was used when the object was created, the value passed to the catalog function must be uppercase because unquoted identifiers are converted to uppercase before being used. If a quoted identifier name was used when the object was created, the value passed to the catalog function must match the case of the name as it was defined. Object names in results returned from catalog functions are returned in the case that they are stored in the database.</td>
</tr>
<tr>
<td></td>
<td>For example, if <strong>Uppercase Identifiers</strong> is set to <strong>ON</strong>, to query the <strong>Account</strong> table you would need to specify:</td>
</tr>
<tr>
<td></td>
<td><code>SELECT &quot;id&quot;, &quot;name&quot; FROM &quot;Account&quot;</code></td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> <strong>ON</strong></td>
</tr>
<tr>
<td>Use Integration Names</td>
<td></td>
</tr>
</tbody>
</table>
The `useIntegrationNames` map option is applicable only to data sources that access Rollbase data for either public cloud or private cloud (on-premise) applications. The `useIntegrationNames` parameter defines the type of name that Hybrid Data Pipeline uses for objects and fields. Every object in Rollbase has a singular name, a plural name, and an integration name. Every field in Rollbase has display name and an integration name. By default, when the map is generated, Hybrid Data Pipeline uses the singular name to generate the table names and the field's display name when generating the column names.

Hybrid Data Pipeline must use the integration names when communicating to Rollbase through the REST API.

To control the object and column names that Hybrid Data Pipeline uses when communicating to Rollbase, enable `useIntegrationName` in the Set Map Options section of the Mapping tab of your data source definition.

**Valid Values:**

| 0 | 1 |

If set to 1, Hybrid Data Pipeline uses the integration names to generate the table and column names.

If set to 0, Hybrid Data Pipeline uses the singular name to generate the table names and the field's display name when generating the column names when the map is generated.

The default value for `useIntegrationNames` is 0.
Advanced tab

Create Rollbase Data Source

Table 83: Advanced tab connection parameters for Progress Rollbase

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption Method</td>
<td>Specifies whether SSL is used to communicate with the Rollbase Web Service. When SSL is enabled, the default, the driver uses the &quot;https&quot; scheme. When SSL is disabled, the driver uses the &quot;http&quot; scheme. The default value is SSL.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Extended Options       | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
  `Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]`  
  If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  
  **Valid Values:** `string`  
  **Default:** empty string |
| Initialization String  | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
  **Syntax:**  
  `command[; command]...`  
  **Where:**  
  `command`  
  is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of `SFORCE`:  
  `InitializationString=(REFRESH SCHEMA SFORCE)`  
  The default is an empty string. |
| Login Timeout          | The amount of time, in seconds, to wait for a connection to be established before timing out the connection request.  
  When set to 0, the connection request never times out.  
  The default value is 0. |
| Max Pooled Statements  | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.  
  The default value of 0 means that the internal prepared statement pooling is not enabled. |
| Read Only              | Sets the connection to read-only mode, that is, the data store can be read but not updated.  
  The default value is `OFF`. |
| Web Service Call Limit | The maximum number of Web service calls allowed to the cloud data store for a single SQL statement or metadata query. |
### Field

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Retry Count</td>
<td>Controls the number of times to retry a Select request that times out. call. Insert, Update, Delete requests are never retried. If set to 0, no retry attempts are made for Select requests that time out after the initial unsuccessful attempt. Valid values are from 0 and any positive integer. The default value is 3.</td>
</tr>
<tr>
<td>Web Service Timeout</td>
<td>The maximum number of Web service calls allowed to the data store for a single SQL statement or metadata query. The value of 0 implies there is no limit. The default value is 120.</td>
</tr>
<tr>
<td>Metadata Exposed Schemas</td>
<td>Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.</td>
</tr>
</tbody>
</table>

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

- `<schema>`

**Where:**

- `<schema>`

  is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

### See also

- [How to create a data source in the Web UI](#)
- [Editing connection parameters](#)

### Salesforce (and Related Data Store) connection parameters

The data source parameters for connecting to the Salesforce and related data stores are similar. However, for simplicity, because the connection features are not identical, the connection parameters are listed separately.

### Salesforce parameters

The following tables describe parameters available on the tabs of a Salesforce.com® Data Source setup dialog:
Chapter 3: Using Hybrid Data Pipeline

- General tab
- OData tab
- Mapping tab
- Advanced tab

**General tab**

Create Salesforce Data Source

<table>
<thead>
<tr>
<th>General</th>
<th>OData</th>
<th>Mapping</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name *</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User ID</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Password</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Salesforce Login URL</td>
<td>login.salesforce.com</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security Token</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Required Fields
Table 84: General tab connection parameters for Salesforce

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of this set of connection parameters.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials for your Salesforce cloud data store account. Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the cloud data store must grant permission to a user with these credentials to access the data store and the target data.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline account.</td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Salesforce Login URL</td>
<td>The data store URL.</td>
</tr>
<tr>
<td></td>
<td>For example, login.salesforce.com.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> login.salesforce.com</td>
</tr>
<tr>
<td></td>
<td>If set to login.salesforce.com, the production environment is used. If set to test.salesforce.com, the test environment is used.</td>
</tr>
<tr>
<td>Security Token</td>
<td>The security token is required to log in to Salesforce from an untrusted network. Salesforce automatically generates this key. If you do not have the security token, log into your account, go to Setup &gt; My Personal Information &gt; Reset My Security Token. A new token will be sent by e-mail.</td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Create Salesforce Data Source

Table 85: OData tab connection parameters for Salesforce

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
**OData Access URI** | Specifies the base URI for the OData feed to access your data source, for example, `https://hybridpipe.operations.com/api/odata/<DataSourceName>`. You can copy the URI and paste it into your application's OData configuration.

The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.

The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding `/$metadata` to the service root URI.

**Schema Map** | Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.

See [Configuring data sources for OData connectivity and working with data source groups](#) on page 622 for more information.

**Data Source Caching** | Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.

Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.

When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.

**Valid Values:**

When set to 1, session caching is enabled. This provides better performance for production.

When set to 0, session caching is disabled. Use this value when you are configuring the data source.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Page Size** | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** $0 | $n  
where $n is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0 |
| **Refresh Result** | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
**Valid Values:**  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
**Default:** 1 |
| **Inline Count Mode** | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
**Valid Values:**  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
**Default:** 1 |
### Top Mode
Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**
- Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.
- Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.

**Default:** 0

### OData Read Only
Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.

Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.

**Valid Values:**
- **ON**
- **OFF**

When **ON** is selected, OData access is restricted to read-only mode.

When **OFF** is selected, write operations can be performed on the OData service.

**Default:** OFF

---

**Mapping tab**
The default values for advanced mapping fields are appropriate in many cases. However, if your organization wants to strip custom prefixes or enable uppercase identifiers, you might want to change map option settings. Understanding how Hybrid Data Pipeline creates and uses maps will help you choose the appropriate values.

The following table describes the mapping options that apply to Salesforce CRM.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Top Mode       | Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries. **Valid Values:**
- Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.
- Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result. **Default:** 0 |
| OData Read Only| Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled. Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon. **Valid Values:**
- **ON**
- **OFF**

When **ON** is selected, OData access is restricted to read-only mode.

When **OFF** is selected, write operations can be performed on the OData service. **Default:** OFF |

**Note:** Map creation is an expensive operation. In most cases, you will only want to re-create a map if you need to change mapping options.
Create Salesforce Data Source

General | OData | Mapping | Advanced

Map Name

Refresh Map
OFF

Create Mapping

Set Map Options

Map System Column Names

Uppercase Identifiers

Audit Columns

Custom Suffix

Keyword Conflict Suffix

Number Field Mapping

Cancel | TEST | Save
The audit columns added by Hybrid Data Pipeline are:

- IsDeleted
- CreatedById
- CreatedDate
- LastModifiedById
- LastModifiedDate
- SYSTEMMODSTAMP

The following table describes the valid values for the Audit Columns parameter.

### Table 87: Valid values for Audit Columns

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Hybrid Data Pipeline includes all of the audit columns and the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>AuditOnly</td>
<td>Hybrid Data Pipeline adds only the audit columns in its table definitions.</td>
</tr>
<tr>
<td>MasterOnly</td>
<td>Hybrid Data Pipeline adds only the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>None</td>
<td>Hybrid Data Pipeline does not add the audit columns or the MasterRecordId column in its table definitions.</td>
</tr>
</tbody>
</table>

The default value for Audit Columns is **All**.

In a typical Salesforce instance, not all users are granted access to the Audit or MasterRecordId columns. If Audit Columns is set to a value other than None and if Hybrid Data Pipeline cannot include the columns requested, the connection fails and an exception is thrown.
Determine whether the Salesforce table mapping files are to be (re)created.

The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

**Table 88: Valid values for Create Map field**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the Map Name field, the name will be a combination of the <strong>UserName</strong> and <strong>Data Source ID</strong>.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the <strong>UserName</strong> and <strong>Data Source ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to <strong>Force New</strong> indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>
Data stores treat the creation of standard and custom objects differently. Objects you create in your organization are called custom objects, and the objects already created for you by the data store administrator are called standard objects.

When you create custom objects such as tables and columns, the data store appends a custom suffix to the name, (__c), two underscores immediately followed by a lowercase “c” character.

For example, Salesforce will create a table named `emp__c` if you create a new table using the following statement:

```
CREATE TABLE emp (id int, name varchar(30))
```

When you expose external objects, Salesforce appends a __x extension (__x), two underscores immediately followed by a lowercase “x” character. This extension is treated in the same way as the __c extension for custom object.

You might expect to be able to query the table using the name you gave it, `emp` in the example. Therefore, by default, the connectivity service strips off the suffix, allowing you to make queries without adding the suffix “__c” or “__x”. The Map Options field allows you to specify a value for CustomSuffix to control whether the map includes the suffix or not:

- If set to include, the map uses the “__c” or “__x” suffix; you must therefore use it in your queries.
- If set to strip, the suffix in the map is removed in the map. Your queries should not include the suffix when referring to custom fields.

The default value for CustomSuffix is include.

The first time you save and test a connection, a map for that data store is created. Once a map is created, you cannot change the map options for that Data Source definition unless you also create a new map. For example, if a map is created with Custom Suffix set to include and then later, you change the Custom Suffix value to strip, you will get an error saying the configuration options do not match. Simply change the value of the Create Map option to force creation of a new map.
The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks.

For example, the Case object is a standard object present in most Salesforce organizations but CASE is also an SQL keyword. Therefore, a table named Case cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:

- Execution of the SQL query `Select * from Case` will return the following:
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce] Unexpected token: CASE in statement [select * from case]

- Execution of the SQL query `Select * from "Case"` will return the following:
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce] Table not found in statement [select * from "Case"]

- Execution of the SQL query, `Select * from "CASE"` will complete successfully.

To avoid using quotes and uppercase for table or column names that match keywords and reserved words, you can instruct Hybrid Data Pipeline to add a suffix to such names. For example, if Keyword Conflict Suffix is set to TAB, the Case table will be mapped to a table named CASETAB. With such a suffix appended in the map, the following queries both work:

- `Select * From CASETAB`
- `Select * From casetab`

### Keyboard Conflict Suffix

Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map.

If you want to name the map yourself, enter a unique name.
By default, when mapping Salesforce system fields to columns in a table, Hybrid Data Pipeline changes system column names to make it evident that the column is a system column. System columns include those for name and id. If the system column names are not changed and you create a new table with id and name columns, the map will need to append a suffix to your columns to differentiate them from the system columns, even if the map option is set to strip suffixes.

If you do not want to change the names of system columns, set this parameter to 0.

Valid values are described in the following table.

Table 89: Valid values for Map System Column Names

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Hybrid Data Pipeline does not change the names of the Salesforce system columns.</td>
</tr>
<tr>
<td>1</td>
<td>Hybrid Data Pipeline changes the names of the Salesforce system columns as described in the following table:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mapped Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>ROWID</td>
</tr>
<tr>
<td>Name</td>
<td>SYS_NAME</td>
</tr>
<tr>
<td>IsDeleted</td>
<td>SYS_ISDELETED</td>
</tr>
<tr>
<td>CreatedDate</td>
<td>SYS_CREATEDDATE</td>
</tr>
<tr>
<td>CreatedById</td>
<td>SYS_CREATEDBYID</td>
</tr>
<tr>
<td>LastModifiedDate</td>
<td>SYS_LASTMODIFIEDDATE</td>
</tr>
<tr>
<td>LastModifeddid</td>
<td>SYS_LASTMODIFIEDID</td>
</tr>
<tr>
<td>SystemModstamp</td>
<td>SYS_SYSTEMMODSTAMP</td>
</tr>
<tr>
<td>LastActivityDate</td>
<td>SYS_LASTACTIVITYDATE</td>
</tr>
</tbody>
</table>

The default value is 0.
In addition to the primitive data types, Hybrid Data Pipeline also defines custom field data types. The **Number Field Mapping** parameter defines how Hybrid Data Pipeline maps fields defined as `NUMBER` (custom field data type). The `NUMBER` data type can be used to enter any number with or without a decimal place.

Hybrid Data Pipeline type casts `NUMBER` data type to the SQL data type `DOUBLE` and stores the values as `DOUBLE`.

This type casting can cause problems when the precision of the `NUMBER` field is greater than the precision of a SQL data type `DOUBLE` value.

By default, Hybrid Data Pipeline maps `NUMBER` values with a precision of 9 or less and scale 0 to the SQL data type `INTEGER` type, and also maps all other `NUMBER` fields to the SQL data type `DOUBLE`. Precision is the number of digits in a number. Scale is the number of digits to the right of the decimal point in a number. For example: The number 123.45 has a precision of 5 and a scale of 2.

Valid values for **Number Field Mapping** are described in the following table.

**Table 90: Valid values for Number Field Mapping**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>alwaysDouble</code></td>
<td>Hybrid Data Pipeline maps <code>NUMBER</code> fields to the SQL data type <code>DOUBLE</code>.</td>
</tr>
<tr>
<td><code>emulateInteger</code></td>
<td>Hybrid Data Pipeline maps <code>NUMBER</code> fields with a precision of 9 or less and scale 0 to the SQL data type <code>INTEGER</code> and maps all other <code>NUMBER</code> fields to the SQL data type <code>DOUBLE</code>.</td>
</tr>
</tbody>
</table>

The default value for **Number Field Mapping** is `emulateInteger`. 

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh Schema</td>
<td>The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>When set to <strong>ON</strong>, the connectivity service attempts to refresh the schema.</td>
</tr>
<tr>
<td></td>
<td>When set to <strong>OFF</strong>, the connectivity service does not attempt to refresh the schema.</td>
</tr>
<tr>
<td>Default</td>
<td><strong>OFF</strong></td>
</tr>
<tr>
<td>Notes</td>
<td>• You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.</td>
</tr>
<tr>
<td></td>
<td>• Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.</td>
</tr>
<tr>
<td></td>
<td>• If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.</td>
</tr>
<tr>
<td>Uppercase Identifiers</td>
<td>Defines how Hybrid Data Pipeline maps identifiers. By default, all unquoted identifier names are mapped to uppercase. Identifiers are object names. Classes, methods, variables, interfaces, and database objects, such as tables, views, columns, indexes, triggers, procedures, constraints, and rules, can have identifiers.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>When set to <strong>ON</strong>, the connectivity service maps all identifier names to uppercase.</td>
</tr>
<tr>
<td></td>
<td>When set to <strong>OFF</strong>, Hybrid Data Pipeline maps identifiers to the mixed case name of the object being mapped. If mixed case identifiers are used, those identifiers must be quoted in SQL statements, and the case of the identifier, must exactly match the case of the identifier name.</td>
</tr>
<tr>
<td>Note</td>
<td>When object names are passed as arguments to catalog functions, the case of the value must match the case of the name in the database. If an unquoted identifier name was used when the object was created, the value passed to the catalog function must be uppercase because unquoted identifiers are converted to uppercase before being used. If a quoted identifier name was used when the object was created, the value passed to the catalog function must match the case of the name as it was defined. Object names in results returned from catalog functions are returned in the case that they are stored in the database.</td>
</tr>
<tr>
<td></td>
<td>For example, if <strong>Uppercase Identifiers</strong> is set to <strong>ON</strong>, to query the <strong>Account</strong> table you would need to specify:</td>
</tr>
<tr>
<td></td>
<td><code>SELECT &quot;id&quot;, &quot;name&quot; FROM &quot;Account&quot;</code></td>
</tr>
<tr>
<td>Default</td>
<td><strong>ON</strong></td>
</tr>
</tbody>
</table>
### Advanced tab

#### Create Salesforce Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Call Limit</td>
<td>The maximum number of Web service calls allowed to the cloud data store for a single SQL statement or metadata query. The default value is 0.</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. Insert, Update, and Delete requests are never retried. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 0.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Web Service Timeout</td>
<td>The time, in seconds, to wait before retrying a timed-out Select request. Valid only if the value of Web Service Retry Count is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application. The default value is 0.</td>
</tr>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, to wait for a connection to be established before timing out the connection request. If set to 0, the connectivity service does not time out a connection request. The default value is 0.</td>
</tr>
<tr>
<td>Enable Bulk Load</td>
<td>Specifies whether to use the bulk load protocol for insert, update, delete, and batch operations. This increases the number of rows that the Hybrid Data Pipeline connectivity service loads to send to the data store. Bulk load reduces the number of network trips. The default value is ON.</td>
</tr>
<tr>
<td>Bulk Load Threshold</td>
<td>Sets a threshold (number of rows) that, if exceeded, triggers bulk loading for insert, update, delete, or batch operations. The default is 4000.</td>
</tr>
<tr>
<td>Enable Bulk Fetch</td>
<td>Specifies whether to use the Salesforce Bulk API for selects based on the value of the Bulk Fetch Threshold option. If the number of rows expected in the result set exceeds the value of Bulk Fetch Threshold, the connectivity service uses the Salesforce Bulk API to execute the select operation. Using the Salesforce Bulk API may significantly reduce the number of Web service calls used to execute a statement and, therefore, may improve performance. The default value is ON.</td>
</tr>
<tr>
<td>Bulk Fetch Threshold</td>
<td>Sets a threshold (number of rows) that, if exceeded, triggers the use of the Salesforce Bulk API for select operations. For this behavior to take effect, the Enable Bulk Fetch option must be set to ON. If set to 0, the Salesforce Bulk API is used for all select operations. The default is 30000 (rows).</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Enable Primary Key Chunking</td>
<td>Specifies whether the driver uses PK chunking for select operations. PK chunking breaks down bulk fetch operations into smaller, more manageable batches for improved performance.</td>
</tr>
<tr>
<td></td>
<td>If set to <strong>ON</strong>, PK chunking is used for select operations when the expected number of rows in the result set is greater than the values of the Bulk Fetch Threshold and Primary Key Chunk Size options. For this behavior to take effect, the Enable Bulk Fetch option must also be set to <strong>ON</strong>.</td>
</tr>
<tr>
<td></td>
<td>If set to <strong>OFF</strong>, PK chunking is not used when executing select operations, and the Primary Key Chunk Size option is ignored.</td>
</tr>
<tr>
<td></td>
<td>The default is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Primary Key Chunk Size</td>
<td>Specifies the size, in rows, of a primary key chunk when PK chunking has been enabled via the Enable Primary Key Chunking option. The Salesforce Bulk API splits the query into chunks of this size.</td>
</tr>
<tr>
<td></td>
<td>Primary Key Chunk Size may be set to a maximum value of 250000 rows.</td>
</tr>
<tr>
<td></td>
<td>The default is 100000 (rows).</td>
</tr>
<tr>
<td>Initialization String</td>
<td>A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.</td>
</tr>
<tr>
<td></td>
<td><strong>Syntax:</strong></td>
</tr>
<tr>
<td></td>
<td><code>command[[; command]...]</code></td>
</tr>
<tr>
<td></td>
<td><strong>Where:</strong></td>
</tr>
<tr>
<td></td>
<td><code>command</code></td>
</tr>
<tr>
<td></td>
<td>is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of <code>SFORCE</code>:</td>
</tr>
<tr>
<td></td>
<td><code>InitializationString=(REFRESH SCHEMA SFORCE)</code></td>
</tr>
<tr>
<td></td>
<td>The default is an empty string.</td>
</tr>
<tr>
<td>Read Only</td>
<td>Sets the connection to read-only mode. Indicates that the cloud data store can be read but not updated.</td>
</tr>
<tr>
<td></td>
<td>The default value is <strong>OFF</strong>.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Extended Options         | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
  - `Database=Server1; UndocumentedOption1=value[; UndocumentedOption2=value;]`  
  If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  
  **Valid Values:** `string`  
  **Default:** empty string  
  **Note:**  
  If you are using a proxy server to connect to your sales cloud instance, then you have to set these options:  
  - `proxyHost = hostname of the proxy server; proxyPort = port number of the proxy server`  
  If Authentication is enabled, then you have to include the following:  
  - `proxyuser=<value>; proxypassword=<value>` |
| Metadata Exposed Schemas | Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.  
  **Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.  
  **Valid Values**  
  - `<schema>`  
  **Where:**  
  - `<schema>`  
  is the name of a valid schema on the backend data store.  
  **Default:** No schema is specified. Therefore, all schemas are exposed.  

**See the steps for:**  
How to create a data source in the Web UI on page 225
See also
Salesforce data store reports on page 966
Salesforce-type data types on page 933
Supported SQL and Extensions on page 967
Supported scalar functions on page 940

FinancialForce parameters
The following tables describe parameters available on the tabs of a FinancialForce.com' Data Source setup dialog:

- General tab
- OData tab
- Mapping tab
- Advanced tab

General tab
Create FinancialForce Data Source
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
<td></td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials for your FinancialForce data store account. Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the cloud data store must grant permission to a user with these credentials to access the data store and the target data.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Note: You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline account.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
<td></td>
</tr>
<tr>
<td>Security Token</td>
<td>The security token is required to log in to Salesforce from an untrusted network. Salesforce automatically generates this key. If you do not have the security token, log into your account, go to Setup &gt; My Personal Information &gt; Reset My Security Token. A new token will be sent by e-mail.</td>
<td></td>
</tr>
<tr>
<td>FinancialForce Login URL</td>
<td>The data store URL. <strong>Valid Values:</strong> login.salesforce.com</td>
<td>test.salesforce.com</td>
</tr>
<tr>
<td></td>
<td>If set to login.salesforce.com, the production environment is used. If set to test.salesforce.com, the test environment is used.</td>
<td></td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 93: OData tab connection parameters for FinancialForce

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **Schema Map**  | Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData.  
See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information. |
| **Data Source Caching** | Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query.  
Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes.  
When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source.  
**Valid Values:**  
When set to 1, session caching is enabled. This provides better performance for production.  
When set to 0, session caching is disabled. Use this value when you are configuring the data source.  
**Default:** 1 |
| **Page Size**   | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
**Valid Values:** 0 | n  
where n is an integer from 1 to 10000.  
When set to 0, the server default of 2000 is used.  
**Default:** 0 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Refresh Result        | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change. **Valid Values:**
  - When set to 0, the OData service caches the first page of results.
  - When set to 1, the OData service re-executes the query.
  - **Default:** 1                                                                 |
| Inline Count Mode     | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging. The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large. **Valid Values:**
  - When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.
  - When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.
  - **Default:** 1                                                                 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use <code>$top</code> and <code>$skip</code> for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses <code>$top</code> to limit the size of the result and rarely attempts to get additional entities by combining <code>$top</code> and <code>$skip</code>.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses <code>$top</code> as part of client-driven paging and generally combines <code>$top</code> and <code>$skip</code> to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>**ON</td>
</tr>
<tr>
<td></td>
<td>When <strong>ON</strong> is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When <strong>OFF</strong> is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> <strong>OFF</strong></td>
</tr>
</tbody>
</table>
Mapping tab

Create FinancialForce Data Source

Map Name

Refresh Map

Create Mapping

Map System Column Names

Uppercase Identifiers

Audit Columns

Custom Suffix

Keyword Conflict Suffix

Number Field Mapping

[Image of the mapping tab interface with options and settings]
Table 94: Mapping tab connection parameters for FinancialForce

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit Columns</td>
<td>The audit columns added by Hybrid Data Pipeline are:</td>
</tr>
<tr>
<td></td>
<td>• IsDeleted</td>
</tr>
<tr>
<td></td>
<td>• CreatedById</td>
</tr>
<tr>
<td></td>
<td>• CreatedDate</td>
</tr>
<tr>
<td></td>
<td>• LastModifiedById</td>
</tr>
<tr>
<td></td>
<td>• LastModifiedDate</td>
</tr>
<tr>
<td></td>
<td>• SYSTEMMODSTAMP</td>
</tr>
</tbody>
</table>

The following table describes the valid values for the Audit Columns parameter.

Table 95: Valid values for Audit Columns

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Hybrid Data Pipeline includes all of the audit columns and the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>AuditOnly</td>
<td>Hybrid Data Pipeline adds only the audit columns in its table definitions.</td>
</tr>
<tr>
<td>MasterOnly</td>
<td>Hybrid Data Pipeline adds only the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>None</td>
<td>Hybrid Data Pipeline does not add the audit columns or the MasterRecordId column in its table definitions.</td>
</tr>
</tbody>
</table>

The default value for Audit Columns is All.

In a typical Salesforce instance, not all users are granted access to the Audit or MasterRecordId columns. If Audit Columns is set to a value other than None and if Hybrid Data Pipeline cannot include the columns requested, the connection fails and an exception is thrown.
Determines whether the Salesforce table mapping files are to be (re)created.

The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

**Table 96: Valid values for Create Map field**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the Map Name field, the name will be a combination of the <strong>UserName</strong> and <strong>Data Source ID</strong>.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the <strong>UserName</strong> and <strong>Data Source ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to <strong>Force New</strong> indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>
Data stores treat the creation of standard and custom objects differently. Objects you create in your organization are called custom objects, and the objects already created for you by the data store administrator are called standard objects.

When you create custom objects such as tables and columns, the data store appends a custom suffix to the name, (__c), two underscores immediately followed by a lowercase “c” character.

For example, Salesforce will create a table named emp__c if you create a new table using the following statement:

```sql
CREATE TABLE emp (id int, name varchar(30))
```

When you expose external objects, Salesforce appends a __extension (__x), two underscores immediately followed by a lowercase “x” character. This extension is treated in the same way as the __c extension for custom object.

You might expect to be able to query the table using the name you gave it, emp in the example. Therefore, by default, the connectivity service strips off the suffix, allowing you to make queries without adding the suffix "__c" or "__x". The Map Options field allows you to specify a value for CustomSuffix to control whether the map includes the suffix or not:

- If set to include, the map uses the "__c" or "__x" suffix; you must therefore use it in your queries.
- If set to strip, the suffix in the map is removed in the map. Your queries should not include the suffix when referring to custom fields.

The default value for CustomSuffix is include.

The first time you save and test a connection, a map for that data store is created. Once a map is created, you cannot change the map options for that DataSource definition unless you also create a new map. For example, if a map is created with Custom Suffix set to include and then later, you change the Custom Suffix value to strip, you will get an error saying the configuration options do not match. Simply change the value of the Create Map option to force creation of a new map.
The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks.

For example, the Case object is a standard object present in most Salesforce organizations but CASE is also an SQL keyword. Therefore, a table named Case cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:

- **Execution of the SQL query** Select * from Case **will return the following**:
  
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce]Unexpected token: CASE in statement [select * from case]

- **Execution of the SQL query** Select * from "Case" **will return the following**:
  
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce]Table not found in statement [select * from "Case"]

- **Execution of the SQL query**, Select * from "CASE" **will complete successfully**.

To avoid using quotes and uppercase for table or column names that match keywords and reserved words, you can instruct Hybrid Data Pipeline to add a suffix to such names. For example, if **Keyword Conflict Suffix** is set to TAB, the Case table will be mapped to a table named CASETAB. With such a suffix appended in the map, the following queries both work:

- Select * From CASETAB
- Select * From casetab

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard Conflict Suffix</td>
<td>The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks. For example, the Case object is a standard object present in most Salesforce organizations but CASE is also an SQL keyword. Therefore, a table named Case cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:</td>
</tr>
<tr>
<td>Map Name</td>
<td>Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map. If you want to name the map yourself, enter a unique name.</td>
</tr>
</tbody>
</table>
By default, when mapping Salesforce system fields to columns in a table, Hybrid Data Pipeline changes system column names to make it evident that the column is a system column. System columns include those for name and id. If the system column names are not changed and you create a new table with id and name columns, the map will need to append a suffix to your columns to differentiate them from the system columns, even if the map option is set to strip suffixes.

If you do not want to change the names of system columns, set this parameter to 0.

Valid values are described in the following table.

**Table 97: Valid values for Map System Column Names**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Hybrid Data Pipeline does not change the names of the Salesforce system columns.</td>
</tr>
<tr>
<td>1</td>
<td>Hybrid Data Pipeline changes the names of the Salesforce system columns as described in the following table:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mapped Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>ROWID</td>
</tr>
<tr>
<td>Name</td>
<td>SYS_NAME</td>
</tr>
<tr>
<td>IsDeleted</td>
<td>SYS_ISDELETED</td>
</tr>
<tr>
<td>CreatedDate</td>
<td>SYS_CREATEDDATE</td>
</tr>
<tr>
<td>CreatedById</td>
<td>SYS_CREATIONBYID</td>
</tr>
<tr>
<td>LastModifiedDate</td>
<td>SYS_LASTMODIFIEDDATE</td>
</tr>
<tr>
<td>LastModifiedId</td>
<td>SYS_LASTMODIFIEDID</td>
</tr>
<tr>
<td>SystemModstamp</td>
<td>SYS_SYSTEMMODSTAMP</td>
</tr>
<tr>
<td>LastActivityDate</td>
<td>SYS_LASTACTIVITYDATE</td>
</tr>
</tbody>
</table>

The default value is 0.
In addition to the primitive data types, Hybrid Data Pipeline also defines custom field data types. The **Number Field Mapping** parameter defines how Hybrid Data Pipeline maps fields defined as `NUMBER` (custom field data type). The `NUMBER` data type can be used to enter any number with or without a decimal place.

Hybrid Data Pipeline type casts `NUMBER` data type to the SQL data type `DOUBLE` and stores the values as `DOUBLE`.

This type casting can cause problems when the precision of the `NUMBER` field is greater than the precision of a SQL data type `DOUBLE` value.

By default, Hybrid Data Pipeline maps `NUMBER` values with a precision of 9 or less and scale 0 to the SQL data type `INTEGER` type, and also maps all other `NUMBER` fields to the SQL data type `DOUBLE`. Precision is the number of digits in a number. Scale is the number of digits to the right of the decimal point in a number. For example: The number 123.45 has a precision of 5 and a scale of 2.

Valid values for **Number Field Mapping** are described in the following table.

**Table 98: Valid values for Number Field Mapping**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>alwaysDouble</code></td>
<td>Hybrid Data Pipeline maps <code>NUMBER</code> fields to the SQL data type <code>DOUBLE</code>.</td>
</tr>
<tr>
<td><code>emulateInteger</code></td>
<td>Hybrid Data Pipeline maps <code>NUMBER</code> fields with a precision of 9 or less and a scale of 0 to the SQL data type <code>INTEGER</code> type, and maps all other <code>NUMBER</code> fields to the SQL data type <code>DOUBLE</code>.</td>
</tr>
</tbody>
</table>

The default value for **Number Field Mapping** is `emulateInteger`. 
### Field: Refresh Schema

The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects.

**Valid Values:**
- When set to **ON**, the connectivity service attempts to refresh the schema.
- When set to **OFF**, the connectivity service does not attempt to refresh the schema.

**Default:**

**OFF**

**Notes**

- You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.
- Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.
- If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.

### Field: Uppercase Identifiers

Defines how Hybrid Data Pipeline maps identifiers. By default, all unquoted identifier names are mapped to uppercase. Identifiers are object names. Classes, methods, variables, interfaces, and database objects, such as tables, views, columns, indexes, triggers, procedures, constraints, and rules, can have identifiers.

**Valid Values:**
- When set to **ON**, the connectivity service maps all identifier names to uppercase.
- When set to **OFF**, Hybrid Data Pipeline maps identifiers to the mixed case name of the object being mapped. If mixed case identifiers are used, those identifiers must be quoted in SQL statements, and the case of the identifier, must exactly match the case of the identifier name.

**Note:** When object names are passed as arguments to catalog functions, the case of the value must match the case of the name in the database. If an unquoted identifier name was used when the object was created, the value passed to the catalog function must be uppercase because unquoted identifiers are converted to uppercase before being used. If a quoted identifier name was used when the object was created, the value passed to the catalog function must match the case of the name as it was defined. Object names in results returned from catalog functions are returned in the case that they are stored in the database.

For example, if **Uppercase Identifiers** is set to **ON**, to query the Account table you would need to specify:

```sql
SELECT "id", "name" FROM "Account"
```

**Default:**

**ON**
**Advanced tab**

**Create FinancialForce Data Source**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Call Limit</td>
<td>The maximum number of Web service calls allowed to the cloud data store for a single SQL statement or metadata query. The default value is 0.</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. Insert, Update, and Delete requests are never retried. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 0.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Web Service Timeout | The time, in seconds, to wait before retrying a timed-out `Select` request. Valid only if the value of **Web Service Retry Count** is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.

Max Pooled Statements | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application. The default value is 0.

Login Timeout | The amount of time, in seconds, to wait for a connection to be established before timing out the connection request. If set to 0, the connectivity service does not time out a connection request. The default value is 0.

Enable Bulk Load | Specifies whether to use the bulk load protocol for insert, update, delete, and batch operations. This increases the number of rows that the Hybrid Data Pipeline connectivity service loads to send to the data store. Bulk load reduces the number of network trips. The default value is **ON**.

Bulk Load Threshold | Sets a threshold (number of rows) that, if exceeded, triggers bulk loading for insert, update, delete, or batch operations. The default is 4000.

Enable Bulk Fetch | Specifies whether to use the Salesforce Bulk API for selects based on the value of the Bulk Fetch Threshold option. If the number of rows expected in the result set exceeds the value of Bulk Fetch Threshold, the connectivity service uses the Salesforce Bulk API to execute the select operation. Using the Salesforce Bulk API may significantly reduce the number of Web service calls used to execute a statement and, therefore, may improve performance. The default value is **ON**.

Bulk Fetch Threshold | Sets a threshold (number of rows) that, if exceeded, triggers the use of the Salesforce Bulk API for select operations. For this behavior to take effect, the Enable Bulk Fetch option must be set to **ON**. If set to 0, the Salesforce Bulk API is used for all select operations. The default is 30000 (rows).
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Primary Key Chunking</td>
<td>Specifies whether the driver uses PK chunking for select operations. PK chunking breaks down bulk fetch operations into smaller, more manageable batches for improved performance. If set to <strong>ON</strong>, PK chunking is used for select operations when the expected number of rows in the result set is greater than the values of the Bulk Fetch Threshold and Primary Key Chunk Size options. For this behavior to take effect, the Enable Bulk Fetch option must also be set to <strong>ON</strong>. If set to <strong>OFF</strong>, PK chunking is not used when executing select operations, and the Primary Key Chunk Size option is ignored. The default is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Primary Key Chunk Size</td>
<td>Specifies the size, in rows, of a primary key chunk when PK chunking has been enabled via the Enable Primary Key Chunking option. The Salesforce Bulk API splits the query into chunks of this size. Primary Key Chunk Size may be set to a maximum value of 250000 rows. The default is 100000 (rows).</td>
</tr>
</tbody>
</table>
| Initialization String         | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed. **Syntax:**

\[
\text{command[; command][; ...]}
\]

**Where:**

\[
\text{command}
\]

is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of **SFORCE**:

\[
\text{InitializationString=(REFRESH SCHEMA SFORCE)}
\]

The default is an empty string. |
<p>| Read Only                     | Sets the connection to read-only mode. Indicates that the cloud data store can be read but not updated. The default value is <strong>OFF</strong>. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Extended Options**         | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
  
  Database=Server1; UndocumentedOption1=value[; UndocumentedOption2=value;]  
  
  If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  
  
  **Valid Values:** string  
  
  **Default:** empty string  
  
  **Note:**  
  
  If you are using a proxy server to connect to your sales cloud instance, then you have to set these options:  
  
  proxyHost = hostname of the proxy server; proxyPort = portnumber of the proxy server  
  
  If Authentication is enabled, then you have to include the following:  
  
  proxyuser=<value>; proxypassword=<value> |

| Metadata Exposed Schemas     | Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.  
  
  **Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.  
  
  **Valid Values**  
  
  <schema>  
  
  Where:  
  
  <schema>  
  
  is the name of a valid schema on the backend data store.  
  
  **Default:** No schema is specified. Therefore, all schemas are exposed. |

See the steps for:  

How to create a data source in the Web UI on page 225
See also
Salesforce data store reports on page 966
Salesforce-type data types on page 933
Supported SQL statements and extensions on page 967
Supported scalar functions on page 940

ServiceMax parameters
The following tables describe parameters available on the tabs of a ServiceMax® Data Source setup dialog:

- General tab
- OData tab
- Mapping tab
- Advanced tab

General tab

Create ServiceMax Data Source

<table>
<thead>
<tr>
<th>General</th>
<th>OData</th>
<th>Mapping</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>Description</td>
<td>User ID</td>
<td>Password</td>
</tr>
<tr>
<td>ServiceMax Login URL</td>
<td>Security Token</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Required Fields

Cancel  TEST  Save
Table 100: General tab connection parameters for ServiceMax

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials for your ServiceMax cloud data store account. Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the cloud data store must grant permission to a user with these credentials to access the data store and the target data.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline account.</td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Security Token</td>
<td>The security token is required to log in to Salesforce from an untrusted network. Salesforce automatically generates this key. If you do not have the security token, log into your account, go to Setup &gt; My Personal Information &gt; Reset My Security Token. A new token will be sent by e-mail.</td>
</tr>
<tr>
<td>ServiceMax Login URL</td>
<td>The data store URL.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>login.salesforce.com</td>
</tr>
<tr>
<td></td>
<td>If set to login.salesforce.com, the production environment is used. If set to test.salesforce.com, the test environment is used.</td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see "Formulating queries" under Querying with OData.
Table 101: OData tab connection parameters for ServiceMax

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the DataSource OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. Valid Values: When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. Default: 1</td>
</tr>
</tbody>
</table>
### Page Size

Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the `$top` and `$skip` parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.

**Valid Values:** $0 | n

where $n$ is an integer from 1 to 10000.

When set to 0, the server default of 2000 is used.

**Default:** 0

### Refresh Result

Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.

**Valid Values:**

When set to 0, the OData service caches the first page of results.

When set to 1, the OData service re-executes the query.

**Default:** 1

### Inline Count Mode

Specifies how the connectivity service satisfies requests that include the `$count` parameter when it is set to `true` (for OData version 4) or the `$inlinecount` parameter when it is set to `allpages` (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.

**Valid Values:**

When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.

When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.

**Default:** 1
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>When ON is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When OFF is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
</tbody>
</table>
Mapping tab

Create ServiceMax Data Source

Map Name

Refresh Map

Create Mapping

Map System Column Names

Uppercase Identifiers

Audit Columns

Custom Suffix

Keyword Conflict Suffix

Number Field Mapping

Required Fields
Table 102: Mapping tab connection parameters for ServiceMax

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Audit Columns | The audit columns added by Hybrid Data Pipeline are:  
• IsDeleted  
• CreatedById  
• CreatedDate  
• LastModifiedById  
• LastModifiedDate  
• SYSTEMMODSTAMP |

The following table describes the valid values for the Audit Columns parameter.

Table 103: Valid values for Audit Columns

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Hybrid Data Pipeline includes all of the audit columns and the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>AuditOnly</td>
<td>Hybrid Data Pipeline adds only the audit columns in its table definitions.</td>
</tr>
<tr>
<td>MasterOnly</td>
<td>Hybrid Data Pipeline adds only the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>None</td>
<td>Hybrid Data Pipeline does not add the audit columns or the MasterRecordId column in its table definitions.</td>
</tr>
</tbody>
</table>

The default value for Audit Columns is All.

In a typical Salesforce instance, not all users are granted access to the Audit or MasterRecordId columns. If Audit Columns is set to a value other than None and if Hybrid Data Pipeline cannot include the columns requested, the connection fails and an exception is thrown.
Determine whether the Salesforce table mapping files are to be (re)created.

The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

Table 104: Valid values for Create Map field

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the Map Name field, the name will be a combination of the User Name and Data Source ID.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the User Name and Data Source ID to name the map. Map creation is expensive, so you will likely not want to leave this option set to Force New indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>
Data stores treat the creation of standard and custom objects differently. Objects you create in your organization are called custom objects, and the objects already created for you by the data store administrator are called standard objects.

When you create custom objects such as tables and columns, the data store appends a custom suffix to the name, (__c), two underscores immediately followed by a lowercase “c” character.

For example, Salesforce will create a table named emp__c if you create a new table using the following statement:

```sql
CREATE TABLE emp (id int, name varchar(30))
```

When you expose external objects, Salesforce appends a_x extension (__x), two underscores immediately followed by a lowercase “x” character. This extension is treated in the same way as the __c extension for custom object.

You might expect to be able to query the table using the name you gave it, emp in the example. Therefore, by default, the connectivity service strips off the suffix, allowing you to make queries without adding the suffix "__c" or "__x". The Map Options field allows you to specify a value for CustomSuffix to control whether the map includes the suffix or not:

- If set to include, the map uses the "__c" or "__x" suffix; you must therefore use it in your queries.
- If set to strip, the suffix in the map is removed in the map. Your queries should not include the suffix when referring to custom fields.

The default value for CustomSuffix is include.

The first time you save and test a connection, a map for that data store is created. Once a map is created, you cannot change the map options for that Data Source definition unless you also create a new map. For example, if a map is created with Custom Suffix set to include and then later, you change the Custom Suffix value to strip, you will get an error saying the configuration options do not match. Simply change the value of the Create Map option to force creation of a new map.
The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks.

For example, the `Case` object is a standard object present in most Salesforce organizations but `CASE` is also an SQL keyword. Therefore, a table named `Case` cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:

- Execution of the SQL query `Select * from Case` will return the following:
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce] Unexpected token: CASE in statement [select * from case]

- Execution of the SQL query `Select * from "Case"` will return the following:
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce] Table not found in statement [select * from "Case"]

- Execution of the SQL query, `Select * from "CASE"` will complete successfully.

To avoid using quotes and uppercase for table or column names that match keywords and reserved words, you can instruct Hybrid Data Pipeline to add a suffix to such names. For example, if **Keyword Conflict Suffix** is set to TAB, the `Case` table will be mapped to a table named `CASETAB`. With such a suffix appended in the map, the following queries both work:

- `Select * From CASETAB`
- `Select * From casetab`

### Keyboard Conflict Suffix

Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map.

If you want to name the map yourself, enter a unique name.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard Conflict Suffix</td>
<td>The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks. For example, the <code>Case</code> object is a standard object present in most Salesforce organizations but <code>CASE</code> is also an SQL keyword. Therefore, a table named <code>Case</code> cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:</td>
</tr>
<tr>
<td>Map Name</td>
<td>Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map. If you want to name the map yourself, enter a unique name.</td>
</tr>
</tbody>
</table>
By default, when mapping Salesforce system fields to columns in a table, Hybrid Data Pipeline changes system column names to make it evident that the column is a system column. System columns include those for name and id. If the system column names are not changed and you create a new table with id and name columns, the map will need to append a suffix to your columns to differentiate them from the system columns, even if the map option is set to strip suffixes.

If you do not want to change the names of system columns, set this parameter to 0.

Valid values are described in the following table.

**Table 105: Valid values for Map System Column Names**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Hybrid Data Pipeline does not change the names of the Salesforce system columns.</td>
</tr>
<tr>
<td>1</td>
<td>Hybrid Data Pipeline changes the names of the Salesforce system columns as described in the following table:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mapped Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>ROWID</td>
</tr>
<tr>
<td>Name</td>
<td>SYS_NAME</td>
</tr>
<tr>
<td>IsDeleted</td>
<td>SYS_ISDELETED</td>
</tr>
<tr>
<td>CreatedDate</td>
<td>SYS_CREATEDDATE</td>
</tr>
<tr>
<td>CreatedById</td>
<td>SYS_CREATEDBYID</td>
</tr>
<tr>
<td>LastModifiedDate</td>
<td>SYS_LASTMODIFIEDDATE</td>
</tr>
<tr>
<td>LastModififiedId</td>
<td>SYS_LASTMODIFIEDID</td>
</tr>
<tr>
<td>SystemModstamp</td>
<td>SYS_SYSTEMMODSTAMP</td>
</tr>
<tr>
<td>LastActivityDate</td>
<td>SYS_LASTACTIVITYDATE</td>
</tr>
</tbody>
</table>

The default value is 0.
In addition to the primitive data types, Hybrid Data Pipeline also defines custom field data types. The **Number Field Mapping** parameter defines how Hybrid Data Pipeline maps fields defined as `NUMBER` (custom field data type). The `NUMBER` data type can be used to enter any number with or without a decimal place.

Hybrid Data Pipeline type casts `NUMBER` data type to the SQL data type `DOUBLE` and stores the values as `DOUBLE`.

This type casting can cause problems when the precision of the `NUMBER` field is greater than the precision of a SQL data type `DOUBLE` value.

By default, Hybrid Data Pipeline maps `NUMBER` values with a precision of 9 or less and a scale of 0 to the SQL data type `INTEGER` type, and also maps all other `NUMBER` fields to the SQL data type `DOUBLE`. Precision is the number of digits in a number. Scale is the number of digits to the right of the decimal point in a number. For example: The number 123.45 has a precision of 5 and a scale of 2.

Valid values for **Number Field Mapping** are described in the following table.

**Table 106: Valid values for Number Field Mapping**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alwaysDouble</td>
<td>Hybrid Data Pipeline maps <code>NUMBER</code> fields to the SQL data type <code>DOUBLE</code>.</td>
</tr>
<tr>
<td>emulateInteger</td>
<td>Hybrid Data Pipeline maps <code>NUMBER</code> fields with a precision of 9 or less and a scale of 0 to the SQL data type <code>INTEGER</code> type, and maps all other <code>NUMBER</code> fields to the SQL data type <code>DOUBLE</code>.</td>
</tr>
</tbody>
</table>

The default value for **Number Field Mapping** is `emulateInteger`. 
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh Schema</td>
<td>The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>When set to <strong>ON</strong>, the connectivity service attempts to refresh the schema.</td>
</tr>
<tr>
<td></td>
<td>When set to <strong>OFF</strong>, the connectivity service does not attempt to refresh the schema.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td><strong>OFF</strong></td>
</tr>
<tr>
<td><strong>Notes:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.</td>
</tr>
<tr>
<td></td>
<td>• Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.</td>
</tr>
<tr>
<td></td>
<td>• If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.</td>
</tr>
<tr>
<td>Uppercase Identifiers</td>
<td>Defines how Hybrid Data Pipeline maps identifiers. By default, all unquoted identifier names are mapped to uppercase. Identifiers are object names. Classes, methods, variables, interfaces, and database objects, such as tables, views, columns, indexes, triggers, procedures, constraints, and rules, can have identifiers.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>When set to <strong>ON</strong>, the connectivity service maps all identifier names to uppercase.</td>
</tr>
<tr>
<td></td>
<td>When set to <strong>OFF</strong>, Hybrid Data Pipeline maps identifiers to the mixed case name of the object being mapped. If mixed case identifiers are used, those identifiers must be quoted in SQL statements, and the case of the identifier, must exactly match the case of the identifier name.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>When object names are passed as arguments to catalog functions, the case of the value must match the case of the name in the database. If an unquoted identifier name was used when the object was created, the value passed to the catalog function must be uppercase because unquoted identifiers are converted to uppercase before being used. If a quoted identifier name was used when the object was created, the value passed to the catalog function must match the case of the name as it was defined. Object names in results returned from catalog functions are returned in the case that they are stored in the database.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>For example, if <strong>Uppercase Identifiers</strong> is set to <strong>ON</strong>, to query the <strong>Account</strong> table you would need to specify:</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>SELECT &quot;id&quot;, &quot;name&quot; FROM &quot;Account&quot;</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td><strong>ON</strong></td>
</tr>
</tbody>
</table>
### Advanced tab

#### Create ServiceMax Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Call Limit</td>
<td>The maximum number of Web service calls allowed to the cloud data store for a single SQL statement or metadata query. The default value is 0.</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. Insert, Update, and Delete requests are never retried. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 0.</td>
</tr>
</tbody>
</table>

---

Table 107: Advanced tab connection parameters for ServiceMax
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Timeout</td>
<td>The time, in seconds, to wait before retrying a timed-out Select request. Valid only if the value of Web Service Retry Count is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application. The default value is 0.</td>
</tr>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, to wait for a connection to be established before timing out the connection request. If set to 0, the connectivity service does not time out a connection request. The default value is 0.</td>
</tr>
<tr>
<td>Enable Bulk Load</td>
<td>Specifies whether to use the bulk load protocol for insert, update, delete, and batch operations. This increases the number of rows that the Hybrid Data Pipeline connectivity service loads to send to the data store. Bulk load reduces the number of network trips. The default value is ON.</td>
</tr>
</tbody>
</table>
Specifies whether the driver uses PK chunking for select operations. PK chunking breaks down bulk fetch operations into smaller, more manageable batches for improved performance.

If set to `ON`, PK chunking is used for select operations when the expected number of rows in the result set is greater than the values of the Bulk Fetch Threshold and Primary Key Chunk Size options. For this behavior to take effect, the Enable Bulk Fetch option must also be set to `ON`.

If set to `OFF`, PK chunking is not used when executing select operations, and the Primary Key Chunk Size option is ignored.

The default is `ON`.

Specifies the size, in rows, of a primary key chunk when PK chunking has been enabled via the Enable Primary Key Chunking option. The Salesforce Bulk API splits the query into chunks of this size.

Primary Key Chunk Size may be set to a maximum value of 250000 rows.

The default is 100000 (rows).

A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.

**Syntax:**

```
command[[; command]...]
```

**Where:**

`command`

is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of `SFORCE`:

```
InitializationString=(REFRESH SCHEMA SFORCE)
```

The default is an empty string.

Sets the connection to read-only mode. Indicates that the cloud data store can be read but not updated.

The default value is `OFF`.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Primary Key Chunking</td>
<td>Specifies whether the driver uses PK chunking for select operations. PK chunking breaks down bulk fetch operations into smaller, more manageable batches for improved performance. If set to <code>ON</code>, PK chunking is used for select operations when the expected number of rows in the result set is greater than the values of the Bulk Fetch Threshold and Primary Key Chunk Size options. For this behavior to take effect, the Enable Bulk Fetch option must also be set to <code>ON</code>. If set to <code>OFF</code>, PK chunking is not used when executing select operations, and the Primary Key Chunk Size option is ignored. The default is <code>ON</code>.</td>
</tr>
<tr>
<td>Primary Key Chunk Size</td>
<td>Specifies the size, in rows, of a primary key chunk when PK chunking has been enabled via the Enable Primary Key Chunking option. The Salesforce Bulk API splits the query into chunks of this size. Primary Key Chunk Size may be set to a maximum value of 250000 rows. The default is 100000 (rows).</td>
</tr>
</tbody>
</table>
| Initialization String  | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed. Syntax: command[[; command]...]
Where: command is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of `SFORCE`: InitializationString=(REFRESH SCHEMA SFORCE) The default is an empty string. |
<p>| Read Only              | Sets the connection to read-only mode. Indicates that the cloud data store can be read but not updated. The default value is <code>OFF</code>. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example: Database=Server1; UndocumentedOption1=value[; UndocumentedOption2=value];</td>
</tr>
<tr>
<td></td>
<td>If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> string</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> empty string</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td></td>
<td>If you are using a proxy server to connect to your sales cloud instance, then you have to set these options:</td>
</tr>
<tr>
<td></td>
<td>proxyHost = hostname of the proxy server; proxyPort = portnumber of the proxy server</td>
</tr>
<tr>
<td></td>
<td>If Authentication is enabled, then you have to include the following:</td>
</tr>
<tr>
<td></td>
<td>proxyUser=&lt;value&gt;; proxypassword=&lt;value&gt;</td>
</tr>
</tbody>
</table>

| Metadata Exposed Schemas  | Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema. |
|                           | **Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data. |
|                           | **Valid Values**                                                                                                                                                                                        |
|                           | <schema>                                                                                                                                                                                                 |
|                           | **Where:**                                                                                                                                                                                              |
|                           | <schema>                                                                                                                                                                                                 |
|                           | is the name of a valid schema on the backend data store.                                                                                                                                             |
|                           | **Default:** No schema is specified. Therefore, all schemas are exposed.                                                                                                                             |

**See the steps for:**

How to create a data source in the Web UI on page 225
**See also**
Salesforce data store reports on page 966
Salesforce-type data types on page 933
Supported SQL statements and extensions on page 967
Supported scalar functions on page 940

**Veeva CRM parameters**
The following tables describe parameters available on the tabs of a Veeva® CRM **Data Source** setup dialog:

- **General tab**
- **OData tab**
- **Mapping tab**
- **Advanced tab**

**General tab**

![Create Veeva CRM Data Source](image)
Table 108: General tab connection parameters for Veeva CRM

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>UserId, Password</td>
<td>The login credentials for your Veeva CRM data store account. Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the cloud data store must grant permission to a user with these credentials to access the data store and the target data.</td>
</tr>
<tr>
<td></td>
<td>You can save the Data Source definition without specifying the login credentials. In that case, when you test the Data source connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the Data Source) in addition to the Data Source name and the credentials for the Hybrid Data Pipeline account.</td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye 🙀 icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Veeva CRM Login URL</td>
<td>The data store URL.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>login.salesforce.com</td>
</tr>
<tr>
<td></td>
<td>If set to login.salesforce.com, the production environment is used.</td>
</tr>
<tr>
<td></td>
<td>If set to test.salesforce.com, the test environment is used.</td>
</tr>
<tr>
<td>Security Token</td>
<td>The security token is required to log in to Salesforce from an untrusted network. Salesforce automatically generates this key. If you do not have the security token, log into your account, go to Setup &gt; My Personal Information &gt; Reset My Security Token. A new token will be sent by e-mail.</td>
</tr>
</tbody>
</table>

**OData tab**

The following table describes the controls on the OData tab. For information on using the Configure Schema editor, see Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
Table 109: OData tab connection parameters for Veeva CRM

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
</tbody>
</table>
| OData Access URI  | Specifies the base URI for the OData feed to access your data source, for example, https://hybridpipe.operations.com/api/odata/<DataSourceName>. You can copy the URI and paste it into your application's OData configuration.  

The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root.  

The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding $/metadata to the service root URI. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Schema Map            | Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. 
See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information. |
| Data Source Caching   | Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. 
Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. 
When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. 
**Valid Values:** 
When set to 1, session caching is enabled. This provides better performance for production. 
When set to 0, session caching is disabled. Use this value when you are configuring the data source. 
**Default:** 1 |
| Page Size             | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages. 
**Valid Values:** 0 | \( n \) 
where \( n \) is an integer from 1 to 10000. 
When set to 0, the server default of 2000 is used. 
**Default:** 0 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Refresh Result      | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  

**Valid Values:**

When set to 0, the OData service caches the first page of results.

When set to 1, the OData service re-executes the query.

**Default:** 1                                                                                               |
| Inline Count Mode   | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  

The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the `count(*)` aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the `count(*)` aggregate; however, it may have a longer initial response time for the first page if the result is large.  

**Valid Values:**

When set to 1, the connectivity service runs a separate `count(*)` aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.

When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  

**Default:** 1                                                                                                                                                                                                                   |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>When ON is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When OFF is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
</tbody>
</table>
Mapping tab

Create Veeva CRM Data Source

Map Name

Refresh Map

Create Mapping

Map System Column Names

Uppercase Identifiers

Audit Columns

Custom Suffix

Keyword Conflict Suffix

Number Field Mapping

Cancel  TEST  Save
Table 110: Mapping tab connection parameters for Veeva CRM

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit Columns</td>
<td>The audit columns added by Hybrid Data Pipeline are:</td>
</tr>
<tr>
<td></td>
<td>• IsDeleted</td>
</tr>
<tr>
<td></td>
<td>• CreatedById</td>
</tr>
<tr>
<td></td>
<td>• CreatedDate</td>
</tr>
<tr>
<td></td>
<td>• LastModifiedById</td>
</tr>
<tr>
<td></td>
<td>• LastModifiedDate</td>
</tr>
<tr>
<td></td>
<td>• SYSTEMMODSTAMP</td>
</tr>
</tbody>
</table>

The following table describes the valid values for the Audit Columns parameter.

Table 111: Valid values for Audit Columns

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Hybrid Data Pipeline includes all of the audit columns and the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>AuditOnly</td>
<td>Hybrid Data Pipeline adds only the audit columns in its table definitions.</td>
</tr>
<tr>
<td>MasterOnly</td>
<td>Hybrid Data Pipeline adds only the MasterRecordId column in its table definitions.</td>
</tr>
<tr>
<td>None</td>
<td>Hybrid Data Pipeline does not add the audit columns or the MasterRecordId column in its table definitions.</td>
</tr>
</tbody>
</table>

The default value for Audit Columns is All.

In a typical Salesforce instance, not all users are granted access to the Audit or MasterRecordId columns. If Audit Columns is set to a value other than None and if Hybrid Data Pipeline cannot include the columns requested, the connection fails and an exception is thrown.
The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.

**Table 112: Valid values for Create Map field**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the <strong>Map Name</strong> field, the name will be a combination of the <strong>UserName</strong> and <strong>DataSource ID</strong>.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the <strong>UserName</strong> and <strong>DataSource ID</strong> to name the map. Map creation is expensive, so you will likely not want to leave this option set to <strong>Force New</strong> indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>
Data stores treat the creation of standard and custom objects differently. Objects you create in your organization are called custom objects, and the objects already created for you by the data store administrator are called standard objects.

When you create custom objects such as tables and columns, the data store appends a custom suffix to the name, (__c), two underscores immediately followed by a lowercase “c” character.

For example, Salesforce will create a table named `emp__c` if you create a new table using the following statement:

```sql
CREATE TABLE emp (id int, name varchar(30))
```

When you expose external objects, Salesforce appends a _x extension (__x), two underscores immediately followed by a lowercase “x” character. This extension is treated in the same way as the __c extension for custom object.

You might expect to be able to query the table using the name you gave it, `emp` in the example. Therefore, by default, the connectivity service strips off the suffix, allowing you to make queries without adding the suffix "__c" or "__x". The Map Options field allows you to specify a value for CustomSuffix to control whether the map includes the suffix or not:

- If set to include, the map uses the "__c" or "__x" suffix; you must therefore use it in your queries.
- If set to strip, the suffix in the map is removed in the map. Your queries should not include the suffix when referring to custom fields.

The default value for CustomSuffix is include.

The first time you save and test a connection, a map for that data store is created. Once a map is created, you cannot change the map options for that Data Source definition unless you also create a new map. For example, if a map is created with Custom Suffix set to include and then later, you change the Custom Suffix value to strip, you will get an error saying the configuration options do not match. Simply change the value of the Create Map option to force creation of a new map.
The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks.

For example, the `Case` object is a standard object present in most Salesforce organizations but `CASE` is also an SQL keyword. Therefore, a table named `Case` cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:

- Execution of the SQL query `Select * from Case` will return the following:
  
  ```
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce] Unexpected token: CASE in statement [select * from case]
  ```

- Execution of the SQL query `Select * from "Case"` will return the following:

  ```
  Error: [DataDirect][DDHybrid JDBC Driver][Salesforce] Table not found in statement [select * from "Case"]
  ```

- Execution of the SQL query `Select * from "CASE"` will complete successfully.

To avoid using quotes and uppercase for table or column names that match keywords and reserved words, you can instruct Hybrid Data Pipeline to add a suffix to such names. For example, if **Keyword Conflict Suffix** is set to TAB, the `Case` table will be mapped to a table named `CASETAB`. With such a suffix appended in the map, the following queries both work:

- `Select * From CASETAB`
- `Select * From casetab`

### Keyboard Conflict Suffix

Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map.

If you want to name the map yourself, enter a unique name.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Keyword Conflict Suffix</strong></td>
<td>The SQL standard and Hybrid Data Pipeline both define keywords and reserved words. These have special meaning in context, and may not be used as identifier names unless typed in uppercase letters and enclosed in quotation marks. For example, the <code>Case</code> object is a standard object present in most Salesforce organizations but <code>CASE</code> is also an SQL keyword. Therefore, a table named <code>Case</code> cannot be used in a SQL statement unless enclosed in quotes and entered in uppercase letters:</td>
</tr>
<tr>
<td><strong>Map Name</strong></td>
<td>Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map. If you want to name the map yourself, enter a unique name.</td>
</tr>
</tbody>
</table>
By default, when mapping Salesforce system fields to columns in a table, Hybrid Data Pipeline changes system column names to make it evident that the column is a system column. System columns include those for name and id. If the system column names are not changed and you create a new table with id and name columns, the map will need to append a suffix to your columns to differentiate them from the system columns, even if the map option is set to strip suffixes.

If you do not want to change the names of system columns, set this parameter to 0.

Valid values are described in the following table.

**Table 113: Valid values for Map System Column Names**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Hybrid Data Pipeline does not change the names of the Salesforce system columns.</td>
</tr>
<tr>
<td>1</td>
<td>Hybrid Data Pipeline changes the names of the Salesforce system columns as described in the following table:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mapped Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>ROWID</td>
</tr>
<tr>
<td>Name</td>
<td>SYS_NAME</td>
</tr>
<tr>
<td>IsDeleted</td>
<td>SYS_ISDELETED</td>
</tr>
<tr>
<td>CreatedDate</td>
<td>SYS_CREATEDDATE</td>
</tr>
<tr>
<td>CreatedById</td>
<td>SYS_CREATEDBYID</td>
</tr>
<tr>
<td>LastModifiedDate</td>
<td>SYS_LASTMODIFIEDDATE</td>
</tr>
<tr>
<td>LastModifiedId</td>
<td>SYS_LASTMODIFIEDID</td>
</tr>
<tr>
<td>SystemModstamp</td>
<td>SYS_SYSTEMMODSTAMP</td>
</tr>
<tr>
<td>LastActivityDate</td>
<td>SYS_LASTACTIVITYDATE</td>
</tr>
</tbody>
</table>

The default value is 0.
In addition to the primitive data types, Hybrid Data Pipeline also defines custom field data types. The **Number Field Mapping** parameter defines how Hybrid Data Pipeline maps fields defined as **NUMBER** (custom field data type). The **NUMBER** data type can be used to enter any number with or without a decimal place.

Hybrid Data Pipeline type casts **NUMBER** data type to the SQL data type **DOUBLE** and stores the values as **DOUBLE**.

This type casting can cause problems when the precision of the **NUMBER** field is greater than the precision of a SQL data type **DOUBLE** value.

By default, Hybrid Data Pipeline maps **NUMBER** values with a precision of 9 or less and scale 0 to the SQL data type **INTEGER** type, and also maps all other **NUMBER** fields to the SQL data type **DOUBLE**. Precision is the number of digits in a number. Scale is the number of digits to the right of the decimal point in a number. For example: The number 123.45 has a precision of 5 and a scale of 2.

Valid values for **Number Field Mapping** are described in the following table.

**Table 114: Valid values for Number Field Mapping**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>alwaysDouble</code></td>
<td>Hybrid Data Pipeline maps <strong>NUMBER</strong> fields to the SQL data type <strong>DOUBLE</strong>.</td>
</tr>
<tr>
<td><code>emulateInteger</code></td>
<td>Hybrid Data Pipeline maps <strong>NUMBER</strong> fields with a precision of 9 or less and a scale of 0 to the SQL data type <strong>INTEGER</strong> and maps all other <strong>NUMBER</strong> fields to the SQL data type <strong>DOUBLE</strong>.</td>
</tr>
</tbody>
</table>

The default value for **Number Field Mapping** is `emulateInteger`. 
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh Schema</td>
<td>The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>When set to <strong>ON</strong>, the connectivity service attempts to refresh the schema.</td>
</tr>
<tr>
<td></td>
<td>When set to <strong>OFF</strong>, the connectivity service does not attempt to refresh the schema.</td>
</tr>
<tr>
<td></td>
<td><strong>Default</strong> <strong>OFF</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Notes</strong></td>
</tr>
<tr>
<td></td>
<td>• You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.</td>
</tr>
<tr>
<td></td>
<td>• Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.</td>
</tr>
<tr>
<td></td>
<td>• If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.</td>
</tr>
<tr>
<td>Uppercase Identifiers</td>
<td>Defines how Hybrid Data Pipeline maps identifiers. By default, all unquoted identifier names are mapped to uppercase. Identifiers are object names. Classes, methods, variables, interfaces, and database objects, such as tables, views, columns, indexes, triggers, procedures, constraints, and rules, can have identifiers.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>When set to <strong>ON</strong>, the connectivity service maps all identifier names to uppercase.</td>
</tr>
<tr>
<td></td>
<td>When set to <strong>OFF</strong>, Hybrid Data Pipeline maps identifiers to the mixed case name of the object being mapped. If mixed case identifiers are used, those identifiers must be quoted in SQL statements, and the case of the identifier, must exactly match the case of the identifier name.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> When object names are passed as arguments to catalog functions, the case of the value must match the case of the name in the database. If an unquoted identifier name was used when the object was created, the value passed to the catalog function must be uppercase because unquoted identifiers are converted to uppercase before being used. If a quoted identifier name was used when the object was created, the value passed to the catalog function must match the case of the name as it was defined. Object names in results returned from catalog functions are returned in the case that they are stored in the database.</td>
</tr>
<tr>
<td></td>
<td>For example, if <strong>Uppercase Identifiers</strong> is set to <strong>ON</strong>, to query the <strong>Account</strong> table you would need to specify:</td>
</tr>
<tr>
<td></td>
<td>SELECT &quot;id&quot;, &quot;name&quot; FROM &quot;Account&quot;</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> <strong>ON</strong></td>
</tr>
</tbody>
</table>
**Advanced tab**

Create Veeva CRM Data Source

### Table 115: Advanced tab connection parameters for Veeva CRM

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Call Limit</td>
<td>The maximum number of Web service calls allowed to the cloud data store for a single SQL statement or metadata query. The default value is 0.</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. Insert, Update, and Delete requests are never retried. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 0.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Web Service Timeout</td>
<td>The time, in seconds, to wait before retrying a timed-out <code>Select</code> request. Valid only if the value of <strong>Web Service Retry Count</strong> is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application. The default value is 0.</td>
</tr>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, to wait for a connection to be established before timing out the connection request. If set to 0, the connectivity service does not time out a connection request. The default value is 0.</td>
</tr>
<tr>
<td>Enable Bulk Load</td>
<td>Specifies whether to use the bulk load protocol for insert, update, delete, and batch operations. This increases the number of rows that the Hybrid Data Pipeline connectivity service loads to send to the data store. Bulk load reduces the number of network trips. The default value is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Bulk Load Threshold</td>
<td>Sets a threshold (number of rows) that, if exceeded, triggers bulk loading for insert, update, delete, or batch operations. The default is 4000.</td>
</tr>
<tr>
<td>Enable Bulk Fetch</td>
<td>Specifies whether to use the Salesforce Bulk API for selects based on the value of the Bulk Fetch Threshold option. If the number of rows expected in the result set exceeds the value of Bulk Fetch Threshold, the connectivity service uses the Salesforce Bulk API to execute the select operation. Using the Salesforce Bulk API may significantly reduce the number of Web service calls used to execute a statement and, therefore, may improve performance. The default value is <strong>ON</strong>.</td>
</tr>
<tr>
<td>Bulk Fetch Threshold</td>
<td>Sets a threshold (number of rows) that, if exceeded, triggers the use of the Salesforce Bulk API for select operations. For this behavior to take effect, the Enable Bulk Fetch option must be set to <strong>ON</strong>. If set to 0, the Salesforce Bulk API is used for all select operations. The default is 30000 (rows).</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Enable Primary Key Chunking  | Specifies whether the driver uses PK chunking for select operations. PK chunking breaks down bulk fetch operations into smaller, more manageable batches for improved performance.  
If set to ON, PK chunking is used for select operations when the expected number of rows in the result set is greater than the values of the Bulk Fetch Threshold and Primary Key Chunk Size options. For this behavior to take effect, the Enable Bulk Fetch option must also be set to ON.  
If set to OFF, PK chunking is not used when executing select operations, and the Primary Key Chunk Size option is ignored.  
The default is ON.                                                                 |
| Primary Key Chunk Size        | Specifies the size, in rows, of a primary key chunk when PK chunking has been enabled via the Enable Primary Key Chunking option. The Salesforce Bulk API splits the query into chunks of this size.  
Primary Key Chunk Size may be set to a maximum value of 250000 rows.  
The default is 100000 (rows).                                                                 |
| Initialization String         | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
**Syntax:**  
command[; command][...]

**Where:**

command

is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:

`InitializationString=(REFRESH SCHEMA SFORCE)`

The default is an empty string.                                                                 |
| Read Only                     | Sets the connection to read-only mode. Indicates that the cloud data store can be read but not updated.  
The default value is OFF.                                                                                                                                 |

---

**Chapter 3: Using Hybrid Data Pipeline**

**Field**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Enable Primary Key Chunking  | Specifies whether the driver uses PK chunking for select operations. PK chunking breaks down bulk fetch operations into smaller, more manageable batches for improved performance.  
If set to ON, PK chunking is used for select operations when the expected number of rows in the result set is greater than the values of the Bulk Fetch Threshold and Primary Key Chunk Size options. For this behavior to take effect, the Enable Bulk Fetch option must also be set to ON.  
If set to OFF, PK chunking is not used when executing select operations, and the Primary Key Chunk Size option is ignored.  
The default is ON.                                                                 |
| Primary Key Chunk Size        | Specifies the size, in rows, of a primary key chunk when PK chunking has been enabled via the Enable Primary Key Chunking option. The Salesforce Bulk API splits the query into chunks of this size.  
Primary Key Chunk Size may be set to a maximum value of 250000 rows.  
The default is 100000 (rows).                                                                 |
| Initialization String         | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
**Syntax:**  
command[; command][...]

**Where:**

command

is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:

`InitializationString=(REFRESH SCHEMA SFORCE)`

The default is an empty string.                                                                 |
| Read Only                     | Sets the connection to read-only mode. Indicates that the cloud data store can be read but not updated.  
The default value is OFF.                                                                                                                                 |
### Extended Options

Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:

```
Database=Server1; UndocumentedOption1=value[; UndocumentedOption2=value;]
```

If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.

**Valid Values:** `string`

**Default:** empty string

**Note:**

If you are using a proxy server to connect to your sales cloud instance, then you have to set these options:

```
proxyHost = hostname of the proxy server; proxyPort = portnumber of the proxy server
```

If Authentication is enabled, then you have to include the following:

```
proxyuser=<value>; proxypassword=<value>
```

### Metadata Exposed Schemas

Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.

**Warning:** This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.

**Valid Values**

```
<schema>
```

**Where:**

```
<schema>
```

is the name of a valid schema on the backend data store.

**Default:** No schema is specified. Therefore, all schemas are exposed.

---

**See the steps for:**

[How to create a data source in the Web UI on page 225](#)
SugarCRM parameters

You define the information that Hybrid Data Pipeline needs to connect to the data store in a data source. These default connection values are used each time you or your application connects to a particular data store. In addition to user credentials, the data store may provide other options you can use to tune performance.

The following tables describes parameter available on the SugarCRM Data Source setup dialog.

- General tab
- Security tab
- Mapping tab
- OData tab
- Advanced tab
**General tab**

**Table 116: General tab connection parameters for SugarCRM**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Connector ID</strong></td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner’s name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td><strong>Data Source Name</strong></td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>User Id, Password</td>
<td>The login credentials for your SugarCRM cloud data store account. Hybrid Data Pipeline uses this information to connect to the data store. The administrator of the cloud data store must grant permission to a user with these credentials to access the data store and the target data.</td>
</tr>
<tr>
<td>Note:</td>
<td>By default, the password is encrypted.</td>
</tr>
<tr>
<td>Note:</td>
<td>You can save the <strong>DataSource</strong> definition without specifying the login credentials. In that case, when you test the <strong>DataSource</strong> connection, you will be prompted to specify these details. Applications using the connectivity service will have to supply the data store credentials (if they are not saved in the <strong>DataSource</strong>) in addition to the <strong>DataSource</strong> name and the credentials for the Hybrid Data Pipeline account.</td>
</tr>
<tr>
<td></td>
<td>By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Specifies the path to the SugarCRM instance. Examples include:</td>
</tr>
<tr>
<td></td>
<td>• <a href="http://localhost/">http://localhost/</a></td>
</tr>
<tr>
<td></td>
<td>• <a href="https://crm.mycompany.com/production/sugarcrm">https://crm.mycompany.com/production/sugarcrm</a></td>
</tr>
<tr>
<td>Default:</td>
<td>None</td>
</tr>
<tr>
<td>OAuth Client ID</td>
<td>Specifies a unique OAuth client Id value for the connection. Each connection must have a unique client Id value. If a second connection is made using the same OAuth client Id, even with another user name, the SugarCRM service may opt to invalidate the access token of the first connection.</td>
</tr>
<tr>
<td>OAuth Client Secret</td>
<td>Specifies the OAuth client shared-secret phrase. The client shared-secret provides credentials between the OAuth server, SugarCRM, and the OAuth client, the Hybrid Data Pipeline connectivity service. SugarCRM supports an empty client secret, although this practice is not recommended.</td>
</tr>
<tr>
<td>OAuth Refresh Token</td>
<td>Specifies the OAuth refresh token value. When used with the clientId and clientSecret, the refresh token provides an alternative method for using OAuth to connect to SugarCRM. In this case, the login behaves just like a relogin, to fetch the access token using the refresh token. If the refresh token is passed, the username and password are ignored, as they are derived from the login the refresh token is associated with.</td>
</tr>
</tbody>
</table>
Creating data sources with the Web UI

Security tab

Create SugarCRM Data Source

- Authentication Method
- Encryption Method

+ Required Fields

[Buttons: Cancel, Test, Save]
Table 117: Security tab connection parameters for SugarCRM

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Method</td>
<td>Determines which authentication method the Hybrid Data Pipeline connectivity service uses when it establishes a connection.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Auto</td>
</tr>
<tr>
<td></td>
<td>If set to Auto, the connectivity service first attempts to use the UserIDPassword method, if sufficient credentials are supplied. If a user ID and password are not specified or are not accepted, the Hybrid Data Pipeline connectivity service tries again using the refreshToken, if supplied. If neither method is successful, the connectivity service throws an exception.</td>
</tr>
<tr>
<td></td>
<td>If set to OAuth, the Hybrid Data Pipeline connectivity service uses only the refresh token method.</td>
</tr>
<tr>
<td></td>
<td>If set to UserIDPassword, the Hybrid Data Pipeline connectivity service uses user ID/password authentication. The connectivity service sends the user ID and password in clear text to the SugarCRM server for authentication. If a user ID and password are not specified, the connectivity service throws an exception.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• The UserId parameter provides the user ID. The Password parameter provides the password. The Encryption Method parameter determines whether the Hybrid Data Pipeline connectivity service uses data encryption.</td>
</tr>
<tr>
<td>Default:</td>
<td>Auto</td>
</tr>
<tr>
<td>Encryption Method</td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the on-premise database server. Note that when using the SugarCRM-hosted version of SugarCRM, as opposed to a locally-installed copy, this will always be SSL, since sugarcrm.com instances always use SSL encryption.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>noEncryption</td>
</tr>
<tr>
<td></td>
<td>If set to noEncryption, data is not encrypted or decrypted.</td>
</tr>
<tr>
<td></td>
<td>If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.</td>
</tr>
<tr>
<td>Default:</td>
<td>SSL</td>
</tr>
</tbody>
</table>
## Mapping tab

### Create SugarCRM Data Source

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Mapping</td>
<td>Determines whether the SugarCRM table mapping files are to be (re)created. The Hybrid Data Pipeline connectivity service automatically maps data store objects and fields to tables and columns the first time that it connects to the data store. The map includes both standard and custom objects and includes any relationships defined between objects.</td>
</tr>
</tbody>
</table>

#### Table 119: Valid values for Create Map field

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not Exist</td>
<td>Select this option for most normal operations. If a map for a data source does not exist, this option causes one to be created. If a map exists, the service uses that existing map. If a name is not specified in the Map Name field, the name will be a combination of the User Name and Data Source ID.</td>
</tr>
<tr>
<td>Force New</td>
<td>Select this option to force creation of a new map. A map is created on connection whether one exists or not. The connectivity service uses a combination of the User Name and Data Source ID to name the map. Map creation is expensive, so you will likely not want to leave this option set to Force New indefinitely.</td>
</tr>
<tr>
<td>No</td>
<td>If a map for a data source does not exist, the connectivity service does not create one.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Map Name | Optional name of the map definition that the Hybrid Data Pipeline connectivity service uses to interpret the schema of the data store. The Hybrid Data Pipeline service automatically creates a name for the map. If you want to name the map yourself, enter a unique name.

Refresh Schema | The Refresh Schema option specifies whether the connectivity service attempts to refresh the schema when an application first connects. 

**Valid Values:**
- When set to **ON**, the connectivity service attempts to refresh the schema.
- When set to **OFF**, the connectivity service does not attempt to refresh the schema.

**Default**
OFF

**Notes**
- You can choose to refresh the schema by clicking the Refresh icon. This refreshes the schema immediately. Note that the refresh option is available only while editing the data source.
- Use the option to specify whether the connectivity service attempts to refresh the schema when an application first connects. Click the Refresh icon if you want to refresh the schema immediately, using an already saved configuration.
- If you are making other edits to the settings, you need to click update to save your configuration. Clicking the Refresh icon will only trigger a runtime call on the saved configuration.

**OData tab**

The following table describes the controls on the **OData** tab. For information on using the **Configure Schema** editor, see Configuring data sources for OData connectivity and working with data source groups on page 622 and Configuring data sources for OData connectivity and working with data source groups on page 622. For information on formulating OData requests, see Formulating queries with OData Version 2 on page 846.
**Table 120: OData tab connection parameters for SugarCRM**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Schema Map</strong></td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See Configuring data sources for OData connectivity and working with data source groups on page 622 for more information.</td>
</tr>
<tr>
<td><strong>Data Source Caching</strong></td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td><strong>PageSize</strong></td>
<td>Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages. <strong>Valid Values:</strong> 0</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Refresh Result    | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
Valid Values:  
When set to 0, the OData service caches the first page of results.  
When set to 1, the OData service re-executes the query.  
Default: 1                                                                                                    |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
Valid Values:  
When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
Default: 1                                                                                                    |
### Top Mode

Indicates how requests typically use `$top` and `$skip` for client side pagination, allowing the service to better anticipate how to process queries.

**Valid Values:**
- Set to 0 when the application generally uses `$top` to limit the size of the result and rarely attempts to get additional entities by combining `$top` and `$skip`.
- Set to 1 when the application uses `$top` as part of client-driven paging and generally combines `$top` and `$skip` to page through the result.

**Default:** 0

### OData Read Only

Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.

Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.

**Valid Values:**
- **ON** | **OFF**

When **ON** is selected, OData access is restricted to read-only mode.

When **OFF** is selected, write operations can be performed on the OData service.

**Default:** OFF
Advanced tab

Create SugarCRM Data Source

- Max Pooled Statements: 0
- Initialization String
- Web Service Call Limit: 0
- Web Service Timeout: 120
- Web Service Retry Count: 2
- Web Service Fetch Size: 0
- Extended Options
- Metadata Exposed Schemas

[Image of Advanced tab interface]
**Table 121: Advanced tab connection parameters for SugarCRM**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Extended Options       | Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:  
  
  Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]  
  
  If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.  
  
  **Valid Values:** string  
  
  **Default:** empty string |
| Initialization String  | A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.  
  
  **Syntax:**  
  
  command[; command]...  
  
  Where:  
  
  command  
  
  is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of SFORCE:  
  
  InitializationString=(REFRESH SCHEMA SFORCE)  
  
  The default is an empty string. |
| Max Pooled Statements  | The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.  
  
  The default value is 0. |
| Web Service Call Limit | The maximum number of Web service calls allowed for a single SQL statement or metadata query.  
  
  When set to 0, there is no limit on the number of Web service calls on a single connection that can be made when executing a SQL statement.  
  
  The default value is 0. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Service Fetch Size</td>
<td>Specifies the number of rows of data the Hybrid Data Pipeline connectivity service attempts to fetch for each call.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td>**0</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the connectivity service attempts to fetch up to a maximum of 10000 rows. This value typically provides the maximum throughput.</td>
</tr>
<tr>
<td></td>
<td>If set to x, the Hybrid Data Pipeline connectivity service attempts to fetch up to a maximum of the specified number of rows. Setting the value lower than 10000 can reduce the response time for returning the initial data. Consider using a smaller value for interactive applications only.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>0</td>
</tr>
<tr>
<td>Web Service Retry Count</td>
<td>The number of times to retry a timed-out Select request. The Web Service Timeout parameter specifies the period between retries. A value of 0 for the retry count prevents retries. A positive integer sets the number of retries. The default value is 2.</td>
</tr>
<tr>
<td>Web Service Timeout</td>
<td>The time, in seconds, to wait before retrying a timed-out Select request. Valid only if the value of <strong>Web Service Retry Count</strong> is greater than zero. A value of 0 for the timeout waits indefinitely for the response to a Web service request. There is no timeout. A positive integer is considered as a default timeout for any statement created by the connection. The default value is 120.</td>
</tr>
<tr>
<td>Metadata Exposed Schemas</td>
<td>Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.</td>
</tr>
<tr>
<td><strong>Warning:</strong></td>
<td>This functionality should not be regarded as a security measure. While the <strong>Metadata Exposed Schemas</strong> option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.</td>
</tr>
<tr>
<td><strong>Valid Values</strong></td>
<td><strong>&lt;schema&gt;</strong></td>
</tr>
<tr>
<td>Where:</td>
<td><strong>&lt;schema&gt;</strong></td>
</tr>
<tr>
<td></td>
<td>is the name of a valid schema on the backend data store.</td>
</tr>
<tr>
<td><strong>Default:</strong></td>
<td>No schema is specified. Therefore, all schemas are exposed.</td>
</tr>
</tbody>
</table>

See the steps for:
How to create a data source in the Web UI on page 225

Sybase parameters

The following tables describe parameters available on the tabs of a Sybase Data Source setup dialog:

- General tab
- Security tab
- OData tab
- Advanced tab

General tab

Create Sybase Data Source

![Create Sybase Data Source](image-url)
Table 122: General tab connection parameters for Sybase

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connector ID</td>
<td>The unique identifier of the On-Premise Connector that is to be used to access the on-premise data source. Select the Connector that you want to use from the dropdown. The identifier can be a descriptive name, the name of the machine where the Connector is installed, or the Connector ID for the Connector. If you have not installed an On-Premise Connector, and no Connectors have been shared with you, this field and drop-down list are empty. If you own multiple Connectors that have the same name, for example, Production, an identifier is appended to each Connector, for example, Production_dup0 and Production_dup1. If the Connectors in the dropdown were shared with you, the owner's name is appended, for example, Production(owner1) and Production(owner2).</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>A unique name for the data source. Data source names can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that is running on the database server.</td>
</tr>
<tr>
<td>Description</td>
<td>A general description of the data source.</td>
</tr>
<tr>
<td>Password</td>
<td>A password for the Sybase account that is used to establish the connection to your Sybase server. By default, the characters in the Password field you type are not shown. If you want the password to be displayed in clear text, click the eye 🔄 icon. Click the icon again to conceal the password.</td>
</tr>
<tr>
<td>Port Number</td>
<td>The port number on which the Sybase database instance is listening for connections.</td>
</tr>
<tr>
<td>Server Name</td>
<td>Specifies either the IP address in IPv4 or IPv6 format, or the server name (if your network supports named servers) of the primary database server, for example, 122.23.15.12 or SybaseAppServer. If using a tnsnames.ora file to provide connection information, do not specify this parameter. Valid Values: string where: string is a valid IP address or server name. The IP address can be specified in either IPv4 or IPv6 format, or a combination of the two.</td>
</tr>
<tr>
<td>User Id</td>
<td>The User Id for the Sybase account used to establish the connection to the Sybase server.</td>
</tr>
</tbody>
</table>
Table 123: Security tab connection parameters for Sybase

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Crypto Protocol Version | Specifies a protocol version or a comma-separated list of the protocol versions that can be used in creating an SSL connection to the data source. If the protocol (or none of the protocols) is not supported by the database server, the connection fails and the connectivity service returns an error.  

**Valid Values:**  
cryptographic_protocol [[, cryptographic_protocol]...]

where:

cryptographic_protocol

  is one of the following cryptographic protocols:

  TLSv1 | TLSv1.1 | TLSv1.2

  The client must send the highest version that it supports in the client hello.  

**Note:** Good security practices recommend using TLSv1.2 if your data source supports that protocol version, due to known vulnerabilities in the earlier protocols.  

**Example**  
Your security environment specifies that you can use TLSv1.1 and TLSv1.2. When you enter the following values, the connectivity service sends TLSv1.2 to the server first.

  TLSv1.1, TLSv1.2

**Default:** TLSv1, TLSv1.1, TLSv1.2
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption Method</td>
<td>Determines whether data is encrypted and decrypted when transmitted over the network between the Hybrid Data Pipeline connectivity service and the database server.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>noEncryption</td>
</tr>
<tr>
<td></td>
<td>If set to noEncryption, data is not encrypted or decrypted.</td>
</tr>
<tr>
<td></td>
<td>If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the Hybrid Data Pipeline connectivity service throws an exception.</td>
</tr>
</tbody>
</table>

**Note:**

- Connection hangs can occur when the Hybrid Data Pipeline connectivity service is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the Login Timeout parameter to avoid problems when connecting to a server that does not support SSL.
- When SSL is enabled, the following properties also apply:
  - HostNameInCertificate
  - ValidateServerCertificate
  - Crypto Protocol Version

**Default:** noEncryption
## Field: Host Name In Certificate

Specifies a host name for certificate validation when SSL encryption is enabled (Encryption Method=SSL) and validation is enabled (Validate Server Certificate=ON). This optional parameter provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server that the Hybrid Data Pipeline connectivity service is connecting to is the server that was requested.

**Valid Values:**

host_name | #SERVERNAME#

where **host_name** is a valid host name.

If **host_name** is specified, the Hybrid Data Pipeline connectivity service compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception.

If **#SERVERNAME#** is specified, the Hybrid Data Pipeline connectivity service compares the server name that is specified in the connection URL or data source of the connection to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the connectivity service compares the host name to the CN part of the certificate’s Subject name. If the values do not match, the connection fails and the connectivity service throws an exception. If multiple CN parts are present, the connectivity service validates the host name against each CN part. If any one validation succeeds, a connection is established.

**Default:** Empty string

## Field: Validate Server Certificate

Determines whether the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server when SSL encryption is enabled (Encryption Method=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA). Allowing the connectivity service to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

**Valid Values:**

ON | OFF

If set to **ON**, the Hybrid Data Pipeline connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file.

If set to **OFF**, the Hybrid Data Pipeline connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any truststore information that is specified by the Java system properties.

**Default:** ON
**OData tab**

The following table describes the controls on the **OData** tab. For information on using the **Configure Schema** editor, see [Configuring data sources for OData connectivity and working with data source groups](#) on page 622. For information on formulating OData requests, see [Formulating queries with OData Version 2](#) on page 846.

![Create Sybase Data Source](image)

**Table 124: OData tab connection parameters for Sybase**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OData Version</td>
<td>Enables you to choose from the supported OData versions. OData configuration made with one OData version will not work if you switch to a different OData version. If you want to maintain the data source with different OData versions, you must create different data sources for each of them.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OData Access URI</td>
<td>Specifies the base URI for the OData feed to access your data source, for example, <a href="https://hybridpipe.operations.com/api/odata/">https://hybridpipe.operations.com/api/odata/</a>&lt;DataSourceName&gt;. You can copy the URI and paste it into your application's OData configuration. The URI contains the case-insensitive name of the data source to connect to, and the query that you want to execute. This URI is the OData Service Root URI for the OData feed. The Service Document for the data source is returned by issuing a GET request to the data source's service root. The OData Service Document returns the names of the entities exposed by the Data Source OData service. To get details such as the properties of the entities exposed, the data types for those properties and the relationships between entities, the Service Metadata Document can be fetched by adding /$metadata to the service root URI.</td>
</tr>
<tr>
<td>Schema Map</td>
<td>Enables OData support. If a schema map is not defined, the OData API cannot be used to access the data store using this data source definition. Use the Configure Schema editor to select the tables/columns to expose through OData. See <a href="#">Configuring data sources for OData connectivity and working with data source groups</a> on page 622 for more information.</td>
</tr>
<tr>
<td>Data Source Caching</td>
<td>Specifies whether the connection to the backend data source is cached in a session associated with the data source. Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be created on every query. Caching of the back end connection can get in the way when trying to configure a data source for OData. If a change is made to any of the Hybrid Data Pipeline data source connection parameters, those changes will not be seen because the connection was established using the old data source definition, and was cached. The session that caches the backend connection is discarded if there is no activity to the data source for approximately 5 minutes. When you configure a data source for OData, it is recommended that the OData session caching be disabled. Once you are satisfied with the OData configuration for the data source, enable the parameter to get the performance improvement provided by caching the connection to the backend data source. <strong>Valid Values:</strong> When set to 1, session caching is enabled. This provides better performance for production. When set to 0, session caching is disabled. Use this value when you are configuring the data source. <strong>Default:</strong> 1</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Page Size    | Determines the number of entities returned on each page for paging controlled on the server side. On the client side, requests can use the $top and $skip parameters to control paging. In most cases, server side paging works well for large data sets. Client side pagination works best with a smaller data sets where it is not as expensive to fetch subsequent pages.  
  **Valid Values:** \( 0 \mid n \)  
  where \( n \) is an integer from 1 to 10000.  
  When set to 0, the server default of 2000 is used.  
  **Default:** 0 |
| Refresh Result | Controls what happens when you fetch the first page of a cached result when using Client Side Paging. Skip must be omitted or set to 0. You can use the cached copy of that first page, or you can re-execute the query to get a new result, discarding the previously cached result. Re-executing the query is useful when the data being fetched may change between two requests for the first page. Using the cached result is useful if you are paging back and forth through results that are not expected to change.  
  **Valid Values:**  
  When set to 0, the OData service caches the first page of results.  
  When set to 1, the OData service re-executes the query.  
  **Default:** 1 |
| Inline Count Mode | Specifies how the connectivity service satisfies requests that include the $count parameter when it is set to true (for OData version 4) or the $inlinecount parameter when it is set to allpages (for OData version 2). These requests require the connectivity service to include the total number of entities that are defined by the OData query request. The count must be included in the first page in server-driven paging and must be included in every page when using client-driven paging.  
  The optimal setting depends on the data store and the size of results. The OData service can run a separate query using the count(*) aggregate to get the count, before running the query used to generate the entities. In very large results, this approach can often lead to the first page being returned faster. Alternatively, the OData service can fetch the entire result before returning the first page. This approach works well for small results and for data stores that cannot optimize the count(*) aggregate; however, it may have a longer initial response time for the first page if the result is large.  
  **Valid Values:**  
  When set to 1, the connectivity service runs a separate count(*) aggregate query to get the count of entities before executing the query to return results. In very large results, this approach can often lead to the first page being returned faster.  
  When set to 2, the connectivity service fetches all entities before returning the first page. For small results, this approach is always faster. However, the initial response time for the first page may be longer if the result is large.  
  **Default:** 1 |
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Mode</td>
<td>Indicates how requests typically use $top and $skip for client side pagination, allowing the service to better anticipate how to process queries.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>Set to 0 when the application generally uses $top to limit the size of the result and rarely attempts to get additional entities by combining $top and $skip.</td>
</tr>
<tr>
<td></td>
<td>Set to 1 when the application uses $top as part of client-driven paging and generally combines $top and $skip to page through the result.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 0</td>
</tr>
<tr>
<td>OData Read Only</td>
<td>Controls whether write operations can be performed on the OData service. Write operations generate a 405 Method Not Allowed response if this option is enabled.</td>
</tr>
<tr>
<td></td>
<td>Existing OData-enabled data sources are read only (write operations are disabled). To enable write operations for an existing OData enabled data source, clear the OData Read Only option on the OData tab. Then, on the Data Sources tab, regenerate the OData model for the data source by clicking on an OData refresh icon.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>**ON</td>
</tr>
<tr>
<td></td>
<td>When ON is selected, OData access is restricted to read-only mode.</td>
</tr>
<tr>
<td></td>
<td>When OFF is selected, write operations can be performed on the OData service.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> OFF</td>
</tr>
</tbody>
</table>
### Advanced tab

#### Table 125: Advanced tab connection parameters for Sybase

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alternate Servers</td>
<td>Specifies one or more alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers connection property.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong> ( (servername1[:port1],[servername2[:port2]]...) )</td>
</tr>
</tbody>
</table>
### Field

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| The server name      | The server name \((\text{servername}_1, \text{servername}_2, \text{and so on})\) is required for each alternate server entry. Port number \((\text{port}_1, \text{port}_2, \text{and so on})\) is optional for each alternate server entry. If the port is unspecified, the port number of the primary server is used. If the port number of the primary server is unspecified, the default port number is used.  
  **Default**: None                                                                 |
| Catalog Options      | Determines which type of metadata information is included in result sets when a JDBC application calls DatabaseMetaData methods. To include multiple types of metadata information, add the sum of the values that you want to include. In this case, specify 6 to include synonyms and to emulate getColumn() calls.  
  **Valid Values**:  
  \(2 \mid 4\)  
  If set to 2, result sets do not contain synonyms.  
  If set to 4, a hint is provided to the Hybrid Data Pipeline connectivity service to emulate getColumn() calls using the ResultSetMetaData object instead of querying database catalogs for column information. Result sets contain synonyms. Using emulation can improve performance because the SQL statement that is formulated by the emulation is less complex than the SQL statement that is formulated using getColumn(). The argument to getColumn() must evaluate to a single table. If it does not, because of a wildcard or null value, for example, the connectivity service reverts to the default behavior for getColumn() calls.  
  **Default**: 2                                                                 |
| Code Page Override   | The code page to be used by the Hybrid Data Pipeline connectivity service to convert Character and Clob data. The specified code page overrides the default database code page or column collation. All Character and Clob data that is returned from or written to the database is converted using the specified code page.  
  By default, the Hybrid Data Pipeline connectivity service automatically determines which code page to use to convert Character data. Use this parameter only if you need to change the connectivity service’s default behavior.  
  **Valid Values**:  
  \(\text{string}\)  
  where \(\text{string}\) is the name of a valid code page that is supported by your JVM. For example, CP950.  
  **Default**: empty string
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Bulk Load</td>
<td>Specifies whether to use the bulk load protocol for insert, update, delete, and batch operations. This increases the number of rows that the Hybrid Data Pipeline connectivity service loads to send to the data store. Bulk load reduces the number of network trips.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td><strong>ON</strong></td>
</tr>
<tr>
<td></td>
<td>If set to <strong>ON</strong>, the Hybrid Data Pipeline connectivity service uses the native bulk load protocols for batch inserts.</td>
</tr>
<tr>
<td></td>
<td>If set to <strong>OFF</strong>, the connectivity service uses the batch mechanism for batch inserts.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> <strong>OFF</strong></td>
</tr>
<tr>
<td>Extended Options</td>
<td>Specifies a semi-colon delimited list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can include any valid connection option in the Extended Options string, for example:</td>
</tr>
<tr>
<td></td>
<td><strong>Database=Server1;UndocumentedOption1=value[;UndocumentedOption2=value;]</strong></td>
</tr>
<tr>
<td></td>
<td>If the Extended Options string contains option values that are also set in the setup dialog, the values of the options specified in the Extended Options string take precedence.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td><strong>string</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> <strong>empty string</strong></td>
</tr>
<tr>
<td>Fetch TWFS AsTime</td>
<td>Determines whether the Hybrid Data Pipeline connectivity service returns column values with the time data type as the JDBC data type TIME or TIMESTAMP.</td>
</tr>
<tr>
<td><strong>Valid Values:</strong></td>
<td><strong>ON</strong></td>
</tr>
<tr>
<td></td>
<td>If set to <strong>ON</strong>, the Hybrid Data Pipeline connectivity service returns column values with the time data type as the JDBC data type TIME. The fractional seconds portion of the value is truncated.</td>
</tr>
<tr>
<td></td>
<td>If set to <strong>OFF</strong>, the Hybrid Data Pipeline connectivity service returns column values with the time data type as the JDBC data type TIMESTAMP. The fractional seconds portion of the value is preserved. Time columns are not searchable when they are described and fetched as timestamp.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> <strong>ON</strong></td>
</tr>
</tbody>
</table>
### Initialization String

A semicolon delimited set of commands to be executed on the data store after Hybrid Data Pipeline has established and performed all initialization for the connection. If the execution of a SQL command fails, the connection attempt also fails and Hybrid Data Pipeline returns an error indicating which SQL commands failed.

**Syntax**:

```
command[; command]...
```

Where:

- `command`

is a SQL command. Multiple commands must be separated by semicolons. In addition, if this property is specified in a connection URL, the entire value must be enclosed in parentheses when multiple commands are specified. For example, assuming a schema name of **SFORCE**:

```
InitializationString=(REFRESH SCHEMA SFORCE)
```

**Default**: empty string

### Load Balancing

Determines whether the connectivity service uses client load balancing in its attempts to connect to the servers (primary and alternate) defined in a Connector group. You can specify one or multiple alternate servers by setting the AlternateServers property.

**Valid Values**: **ON** | **OFF**

- **ON**: The connectivity service uses client load balancing and attempts to connect to the servers (primary and alternate) in random order. The connectivity service randomly selects from the list of primary and alternate On Premise Connectors which server to connect to first. If that connection fails, the connectivity service again randomly selects from this list of servers until all servers in the list have been tried or a connection is successfully established.

- **OFF**: The connectivity service does not use client load balancing and connects to each server based on their sequential order (primary server first, then, alternate servers in the order they are specified).

**Default**: **OFF**

**Notes**

- The Alternate Servers connection parameter specifies one or multiple alternate servers for failover and is required for all failover methods. To turn off failover, do not specify a value for the Alternate Servers parameter.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login Timeout</td>
<td>The amount of time, in seconds, that the Hybrid Data Pipeline connectivity service waits for a connection to be established before timing out the connection request.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values:</strong></td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>where x is a positive integer that represents a number of seconds.</td>
</tr>
<tr>
<td></td>
<td>If set to 0, the Hybrid Data Pipeline connectivity service does not time out a connection request.</td>
</tr>
<tr>
<td></td>
<td>If set to x, the Hybrid Data Pipeline connectivity service waits for the specified number of seconds before returning control to the application and throwing a timeout exception.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> 30</td>
</tr>
<tr>
<td>Max Pooled Statements</td>
<td>The maximum number of prepared statements to cache for this connection. If the value of this property is set to 20, the connectivity service caches the last 20 prepared statements that are created by the application.</td>
</tr>
<tr>
<td></td>
<td>The default value is 0.</td>
</tr>
<tr>
<td>Metadata Exposed Schemas</td>
<td>Restricts the metadata exposed by Hybrid Data Pipeline to a single schema. The metadata exposed in the SQL Editor, the Configure Schema Editor, and third party applications will be limited to the specified schema. JDBC, OData, and ODBC metadata calls will also be restricted. In addition, calls made with the Schema API will be limited to the specified schema.</td>
</tr>
<tr>
<td></td>
<td><strong>Warning:</strong> This functionality should not be regarded as a security measure. While the Metadata Exposed Schemas option restricts the metadata exposed by Hybrid Data Pipeline to a single schema, it does not prevent queries against other schemas on the backend data store. As a matter of best practice, permissions should be set on the backend data store to control the ability of users to query data.</td>
</tr>
<tr>
<td></td>
<td><strong>Valid Values</strong></td>
</tr>
<tr>
<td></td>
<td>&lt;schema&gt;</td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td>&lt;schema&gt;</td>
</tr>
<tr>
<td></td>
<td>is the name of a valid schema on the backend data store.</td>
</tr>
<tr>
<td></td>
<td><strong>Default:</strong> No schema is specified. Therefore, all schemas are exposed.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Query Timeout            | Sets the default query timeout (in seconds) for all statements created by a connection. **Valid Values:**  
  -1 | 0 | x  
  If set to −1, the query timeout functionality is disabled. The Hybrid Data Pipeline connectivity service silently ignores calls to the `Statement.setQueryTimeout()` method.  
  If set to 0, the default query timeout is infinite (the query does not time out).  
  If set to x, the Hybrid Data Pipeline connectivity service uses the value as the default timeout for any statement that is created by the connection. To override the default timeout value set by this connection option, call the `Statement.setQueryTimeout()` method to set a timeout value for a particular statement.  
  **Default:** 0  

| Result Set Meta Data Options | Determines whether the Hybrid Data Pipeline connectivity service returns table name information in the ResultSet metadata for Select statements.  
**Valid Values:**  
0 | 1  
If set to 0 and the `ResultSetMetaData.getTableName()` method is called, the Hybrid Data Pipeline connectivity service does not perform additional processing to determine the correct table name for each column in the result set. The `getTableName()` method may return an empty string for each column in the result set.  
If set to 1 and the `ResultSetMetaData.getTableName()` method is called, the Hybrid Data Pipeline connectivity service performs additional processing to determine the correct table name for each column in the result set. The connectivity service returns schema name and catalog name information when the `ResultSetMetaData.getSchemaName()` and `ResultSetMetaData.getCatalogName()` methods are called if the connectivity service can determine that information.  
**Default:** 0  

| Transaction Mode         | Controls how the connectivity service delimits the start of a local transaction.  
**Valid Values:**  
imPLICIT | explicit  
If set to implicit, the connectivity service uses implicit transaction mode. This means that Sybase, not the connectivity service, automatically starts a transaction when a transactionable statement is executed. Typically, implicit transaction mode is more efficient than explicit transaction mode because the connectivity service does not have to send commands to start a transaction and a transaction is not started until it is needed. When `TRUNCATE TABLE` statements are used with implicit transaction mode, Sybase may roll back the transaction if an error occurs. If this occurs, use the explicit value for this property.  
If set to explicit, the connectivity service uses explicit transaction mode. This means that the connectivity service, not Sybase, starts a new transaction if the previous transaction was committed or rolled back.  
**Default:** implicit  

---

**Chapter 3: Using Hybrid Data Pipeline**

**Progress DataDirect Hybrid Data Pipeline: User’s Guide: Version 4.6**
See the steps for:

How to create a data source in the Web UI on page 225

Editing, testing, and deleting data sources with the Web UI

Hybrid Data Pipeline data sources can be modified, tested, and deleted, as described in the following topics.

Note: While administrators can modify their own data sources with the Web UI, they cannot modify data sources on behalf of users in the Web UI. In addition, administrators cannot set permissions on data sources with the Web UI. To modify data sources on behalf of a user or set permissions on data sources, an administrator must execute API operations with the DataSource API.

Editing connection parameters

Take the following steps to edit a data source definition

1. Navigate to the Data Sources view by clicking the data sources icon.
2. Select the data source you want to edit.
   • Option 1. Click the data source you want to edit from the list of data sources.
   • Option 2. Tick the checkbox of the data source you want to edit. Then, select Edit from the Actions dropdown.
3. Modify the values of parameters under each of the tabs, as desired.
4. Click Update to apply the changes to the data source definition.
5. Click TEST to establish a connection with the data store.

Testing a data source

You can use the SQL Editor to browse data source schema and test data sources by executing SQL queries.

Take the following steps to view a data source and run queries against it.

1. Navigate to the SQL Editor view by clicking the SQL editor icon.
2. From the Select a Data Source dropdown, select the data source you want to view or query.
3. To view schema tables, click the a schema carrot in the Schema Tree panel.

For backend data stores that support schemas, the Metadata Exposed Schemas option can be used to restrict the exposed schemas to a single schema. Metadata Exposed Schemas only affects the metadata that is displayed in the Schema navigation pane. SQL queries can still be executed against tables in other schemas. For details, see the parameters topic for your data source type.
4. To view the details of a table, click on a table in the Table Details panel.

5. To query a data source, enter a SQL query in the Editor or drag the table name, and then click EXECUTE to run the query.

The results of the query are displayed in the Results section along with the status of the query execution. The maximum number of rows displayed per query is 200.

Deleting a data source

⚠️ Warning: Once a data source is deleted, you cannot undo the delete action.

⚠️ Warning: Deleting a data source that is a member of a data source group affects the configuration of the data source group. See Configuring data sources for OData connectivity and working with data source groups on page 622 for details.

Take the following steps to delete a data source.

1. Navigate to the Data Sources view by clicking the data sources icon.
2. Tick the checkbox of the data source you want to delete (you may select more than one). Then, select Delete from the Actions dropdown.
   A confirmation dialog appears.
3. Click DELETE to delete the data source.

Configuring data sources for OData connectivity and working with data source groups

Hybrid Data Pipeline supports OData Version 2 and Version 4 connectivity for all supported data stores. When creating a data source on a backend data store, OData access can be enabled and configured on the OData tab of any data store in the Web UI. As part of the process for creating an OData-enabled data source, you must configure an OData schema map with the Configure Schema editor. (The editor may be accessed by clicking Configure and navigating to the OData tab.) The schema map that you configure exposes the backend data in an OData model. Once the schema map has been configured and the data source saved, OData queries can be made to the data source the URL provided in the OData Access URI field.

In some cases, you might want to access multiple OData schemas with the same resource path. This can be achieved by creating a data source group of OData-enabled data sources. You begin this process by creating individual OData-enabled data sources. These data sources can be created on one or more backend data stores. Once the OData-enabled data sources have been created, you can proceed by creating a data source group comprised of these data sources. The URI provided in the OData Access URI field of the data source group can then be used in your OData resource path.

Note: OData Version 4 applications and services should be used in application environments, instead of OData Version 2, whenever possible. OData Version 4 provides enhancements and advanced features that are not available in OData Version 2.
The topics in this section provide details on enabling OData connectivity and working with data source groups.

- Configuring data sources for OData Version 2 connectivity on page 623
- Configuring data sources for OData Version 4 connectivity on page 627
- Creating a data source group on page 635
- Editing a data source group on page 636
- Deleting a data source group on page 637

See also
- Supported data stores on page 226
- Getting started with OData Version 2 on page 827
- Getting started with OData Version 4 on page 863

Configuring data sources for OData Version 2 connectivity

Hybrid Data Pipeline supports OData Version 2 and Version 4 connectivity for all supported data stores. You can configure a data source on any data store for OData connectivity either during the process of creating the data source or after the data source has been created.

The following steps describe how to configure a data source for OData Version 2 connectivity.

1. From the Web UI, navigate to the Data Sources view by clicking the data sources icon.
   - **Option 1.** If creating a new data source, click New Data Source, choose the data store, enter the required information on the General tab, and click TEST to confirm connectivity to the backend data store. (See Creating data sources with the Web UI on page 224 for details.)
   - **Option 2.** If enabling OData on an existing data source, select the data source you wish to modify.

2. Select the OData tab.
3. For **OData Version**, select **Version 2**.

4. Open the Configure Schema editor by clicking **Configure** to the right of the **Schema Map** field.

![Configure Schema Editor](image-url)

5. Select a schema from the **Select Schema** dropdown.

**Note:** By default, Hybrid Data Pipeline exposes all schemas on any backend data stores that support multiple schemas. The **Metadata Exposed Schemas** option on the **Advanced** tab for any such data store can be used to limit exposed schemas to a single schema. If a schema is selected for the **Metadata Exposed Schemas** option, it will be the only schema available on the Configure Schema editor’s **Select Schema** dropdown.

6. Select the **Tables and Columns** tab. Then select and define the tables and columns you want to expose to OData client applications.

- To add all tables, click **Add All Tables** on the **Tables** panel.
- To add individual tables, select a table on the **Tables** panel and click **Add To Map** in the **Settings** panel to the right.
- To remove a table that was previously added, select the table and click **Remove From Map** in the **Settings** panel.
- To specify singular and plural alias names for a table, select the table, enter the table alias for the entity type name in the **Singular Name** field, enter the table alias for the entity collection name in the **Plural Name** field, and click **Add To Map**.

**Note:** The singular alias name specified is used as the entity type name, while the plural alias name will be used as the entity collection name. If alias names are not specified, the table name is used as the entity type name and pluralized for the entity collection name. For example, the entity type name for the table **ACCOUNTS** would be **ACCOUNTS**, while the entity collection name would be **ACCOUNTSES**.

- To specify a column as a primary key, select the column from the **Columns** panel and set the **Is Primary Key** switch from **OFF** to **ON**.
Note: The Configure Schema editor indicates that a primary key exists for a table with a skeleton key icon. A primary key assigned in the backend data store cannot be changed. If a primary key has not been discovered for a table you wish to map, one or more columns must be specified as a primary key.

- To remove a column from the OData schema map, select the column from the Columns panel and click Remove From Map in the Settings panel.

Note: When a table is added, all columns in the table are exposed in the OData schema map by default. You can modify the columns exposed by removing (or excluding) them from the schema map.

7. Take the following steps to enable text search for individual tables and text-based columns using the ddsearch custom query parameter.
   a) Select a table from the Tables panel.
   b) Specify a search option from the Search Options dropdown. Then click Add To Map.
      - Full Text is only available for data store types that support indexing and full text search.
      - Substring enables searches for the string anywhere in the search-enabled fields.
      - Begins restricts the search to the text at the beginning of a field.
   c) If you selected Full Text in Step b, you should select an index type for all text-based columns. Select the column from the Columns panel, and specify an index type from the Index Type dropdown in the Settings panel. Then click Add To Map.
      The index type is the type of index supported by the backend data store. TEXT is the only valid value for the DB2 and SQL Server data stores. CONTEXT and CTXCAT are the valid values for the Oracle data store. If Full Text has been selected but the data store index has not been properly configured, queries using ddsearch will return errors.
   d) If you selected Substring or Begins in Step b, you should select which text-based columns can be searched. Select the column from the Columns panel, and set the Is Searchable switch to ON. Then click Add To Map.

8. Click the Review Schema Map tab to review the OData schema map in JSON format.
9. Click **Save Map** to save your configuration of the OData schema map.

10. Set OData options to the desired values.

   - **Data Source Caching** controls caching of results. A value of 0 results in a stateless session, requiring Hybrid Data Pipeline to access the data store for each request. A value of 1, the default, allows Hybrid Data Pipeline to cache results, improving performance for repeated requests for the same entity. Hybrid Data Pipeline clears the cache after ten minutes of inactivity or at the end of a session.

   - **Page Size** controls the number of results returned in one response. By default, the value in this field is 0 which causes Hybrid Data Pipeline to return up to 2,000 top-level entities per response. If the response contains more than 2,000 entities, the first 2,000 entities are returned and the end of the response contains a link that the OData client can use to fetch the next set. You can set the page size by using values from 1 to 10,000. Client requests can also specify the size of results with query parameters.

   - **Refresh Result** determines whether Hybrid Data Pipeline returns results from the cache (for entities in the cache) or queries the data source again. A value of 1, the default, allows Hybrid Data Pipeline to satisfy requests from cached results. A value of 0 forces queries to the backend data store. If caching is not enabled, this parameter has no effect.

   - **Inline Count Mode** controls how Hybrid Data Pipeline handles requests that include the $inlinecount parameter with a value of allpages. The response includes the total number of entities that satisfy the query. A value of 0 causes Hybrid Data Pipeline to skip counting. A value of 1 causes Hybrid Data Pipeline to run a separate query to get the count before the query that returns the entities. This can result in the first page of results being returned faster for large result sets for some data store types. A value of 2, the default, causes Hybrid Data Pipeline to fetch all results and calculate the total number before returning the first page of results to the client.

   - **Top Mode** allows Hybrid Data Pipeline to better handle requests that include the $top parameter. A value of 0, the default, indicates that clients using $top to limit result set size will rarely attempt to get additional entities using the $skip parameter. A value of 1 indicates that clients generally use $top and $skip together to paginate results.

   - **OData Read Only** controls read/write access. For a new data source definition, this option is not selected by default. For a data source definition where OData was enabled before this option was available, it will be checked by default. Remove the check mark to enable write access.

11. Click **Update** to save your work.

What to do next:
Test your OData-enabled data source as described in Testing data source configurations (OData Version 2) on page 833.

After you create an OData-enabled data source, you can view the status of the schema map generation on the DataSources screen. The icon besides the OData-enabled data source indicates the status of the schema map generation. The following table provides details of the icons.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Icon 1]</td>
<td>The synchronization of the schema map is in progress. The number denotes the percentage of synchronization completed.</td>
</tr>
<tr>
<td>![Icon 2]</td>
<td>The schema map was synchronized successfully.</td>
</tr>
<tr>
<td>![Icon 3]</td>
<td>The schema map was synchronized successfully, but there are some table/column warnings. Hybrid Data Pipeline allows users to know the details of the tables/columns and/or functions that were dropped while generating the OData Model for a given schema map of a Data Source. The number of warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the Schema API on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td>![Icon 4]</td>
<td>Errors occurred while synchronizing the schema map. You must address the errors and synchronize the schema map again. Hybrid Data Pipeline allows users to know the details of the tables and/or columns that were dropped while generating the OData Model for a given schema map of a Data Source. The number of errors/warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the Schema API on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td>![Icon 5]</td>
<td>You must synchronize the schema map again.</td>
</tr>
</tbody>
</table>

Configuring data sources for OData Version 4 connectivity

Hybrid Data Pipeline supports OData Version 2 and Version 4 connectivity for all supported data stores. You can configure a data source on any data store for OData connectivity either during the process of creating the data source or after the data source has been created.

The following steps describe how to configure a data source for OData Version 4 connectivity.

1. From the Web UI, navigate to the Data Sources view by clicking the data sources icon.
   - **Option 1.** If creating a new data source, click New Data Source, choose the data store, enter the required information on the General tab, and click TEST to confirm connectivity to the backend data store. (See Creating data sources with the Web UI on page 224 for details.)
• **Option 2.** If enabling OData on an existing data source, select the data source you wish to modify.

2. Select the **OData** tab.

3. For **OData Version**, select **Version 4**.

4. Open the Configure Schema editor by clicking **Configure** to the right of the **Schema Map** field.

5. Select a schema from the **Select Schema** dropdown.
Note: By default, Hybrid Data Pipeline exposes all schemas on any backend data stores that support multiple schemas. The Metadata Exposed Schemas option on the Advanced tab for any such data store can be used to limit exposed schemas to a single schema. If a schema is selected for the Metadata Exposed Schemas option, it will be the only schema available on the Configure Schema editor’s Select Schema dropdown.

6. From the Tables and Columns tab, select and define the tables and columns you want to expose to OData client applications.

   • To add all tables, click Add All Tables on the Tables panel.
   • To add individual tables, select a table on the Tables panel and click Add To Map in the Settings panel to the right.
   • To remove a table that was previously added, select the table and click Remove From Map in the Settings panel.
   • To specify singular and plural alias names for a table, select the table, enter the table alias for the entity type name in the Singular Name field, enter the table alias for the entity collection name in the Plural Name field, and click Add To Map.

   Note: The singular alias name specified is used as the entity type name, while the plural alias name will be used as the entity collection name. When alias names are not specified, the mapping of entity names will be dictated by the Entity Name Mode setting in the OData Settings tab, as described in Step 9.

   • To specify a column as a primary key, select the column from the Columns panel and set the Is Primary Key switch from OFF to ON.

   Note: The Configure Schema editor indicates that a primary key exists for a table with a skeleton key icon. A primary key assigned in the backend data store cannot be changed. If a primary key has not been discovered for a table you wish to map, one or more columns must be specified as a primary key.

   • To remove a column from the OData schema map, select the column from the Columns panel and click Remove From Map in the Settings panel.

   Note: When a table is added, all columns in the table are exposed in the OData schema map by default. You can modify the columns exposed by removing (or excluding) them from the schema map.

7. From the Tables and Columns tab, select the columns you want to view or modify.
• To specify an alias name for a column, select the column and enter an alias in the **Alias Name** field. If specified, the alias name will be used as the OData name for the column. If not specified, the name of the column will be used as the OData name.

• To specify a column as a primary key, set the **Is Primary Key** switch from **OFF** to **ON**.

**Note:** The Configure Schema editor indicates that a primary key exists for a table with a skeleton key icon. A primary key assigned in the backend data store cannot be changed. If a primary key has not been discovered for a table you wish to map, one or more columns must be specified as a primary key.

• Open **Advanced Settings** to review and modify column metadata. The **Advanced Settings** allow you to modify column metadata returned by the underlying JDBC driver. This is especially useful when the JDBC driver returns incorrect metadata. The **Driver Value** of each setting indicates the value that is returned by the driver. You can specify settings related to the following properties:

  • **Data Type**: Indicates the data type for the column. If you wish to use the **Actual Value**, you can leave the Data Type as **Default**. If you wish to override the data type specified, you can choose an alternate data type from the dropdown list.

  **Note:** Depending on the data types selected, some of the Advanced settings options will be enabled or disabled. For example, **Scale** is enabled for the decimal dataype, and not for the integer datatype.

  • **Column Size or Precision**: Indicates the maximum precision or maximum length of the column.

  • **Scale**: Indicates the maximum scale of the column.
• **Is Nullable**: Indicates whether the column can have a null value. Normally drivers report this correctly. Some drivers may report a column as not nullable while null values exist in the column. In such a scenario, the *is Nullable* could be set to true to correct this issue. Note that there could be implications on the create entity behavior by changing this setting.

• **Is Auto Increment**: Indicates whether the column is a uniquely generated column. Setting this to true will indicate to the service that it should ignore incoming values for this column during the create, update, and patch entity operations.

• **Is Generated**: Indicates whether the column is a generated value. If the column is generated, then the OData code will ignore incoming values for this column during the create, update, and patch entity operations.

8. Take the following steps to enable text search for individual tables and text-based columns using the `$search` system query option.

   a) Select a table from the **Tables** panel.

   b) Specify a search option from the **Search Options** dropdown. Then click **Add To Map**.

      • **Full Text** is only available for data store types that support indexing and full text search.

      • **Substring** enables searches for the string anywhere in the search-enabled fields.

      • **Begins** restricts the search to the text at the beginning of a field.

   c) If you selected **Full Text** in Step b, you should select an index type for all text-based columns. Select the column from the **Columns** panel, and specify an index type from the **Index Type** dropdown in the **Settings** panel. Then click **Add To Map**.

      The index type is the type of index supported by the backend data store. **TEXT** is the only valid value for the DB2 and SQL Server data stores. **CONTEXT** and **CTXCAT** are the valid values for the Oracle data store. If **Full Text** has been selected but the data store index has not been properly configured, queries using `$search` will return errors.

   d) If you selected **Substring** or **Begins** in Step b, you should select which text-based columns can be searched. Select the column from the **Columns** panel, and set the **Is Searchable** switch to **ON**. Then click **Add To Map**.

9. Take the following steps to expose stored functions. (Note that stored functions are supported only for DB2, SQL Server, and Oracle data stores. See **Stored functions support** on page 880 for details on further restrictions).
a) Select the **Functions** tab.
b) Select the function you want to expose from the **Functions** panel.
c) If desired, specify an alias name for the stored function.
d) If desired, specify an import alias name for a function import that corresponds to the function.
e) Specify whether the OData type is a function or an action on the **OData Type** dropdown.
f) Click **Add To Map**.

10. Specify general settings on the **OData Settings** tab. Then click **Add To Map** to apply settings.

- From the **Entity Name Mode** dropdown, specify the algorithm used to map table names to entity collection names or entity type names. Entity collection names are usually plural, while entity type names are usually singular.
• When **guess** (default) is selected, one of the following algorithms is applied based on an evaluation of the table name.
  
  • If the table name ends with a numeric digit, the table name is used as the entity collection name and a suffix is appended to the table name for the entity type name. The suffix used can be specified in the **Singular Suffix** field.
  
  • If the table name does not end with a digit and appears to be singular, the table name is used as the entity collection name and singularized for the entity type name.
  
  • If the table name does not end with a digit and appears to be plural, the table name is used as the entity type name and pluralized for the entity collection name.
  
  • When **singularize** is selected, the table name is used as the entity collection name. The table name is then singularized for the entity type name.
  
  • When **pluralize** is selected, the table name is used as the entity type name. The table name is then pluralized for the entity collection name.
  
  • When **suffix** is selected, the table name is used as the entity collection name. For the entity type name, a suffix is appended to the table name. The suffix used can be specified in the **Singular Suffix** field.
  
  • With the **Time As String** switch, specify how the JDBC type Time should be mapped.
    
    • If set to **OFF** (default), Time is mapped to the OData type TimeOfDay.
    
    • If set to **ON**, Time is mapped as String.
  
  • In the **Singular Suffix** field, enter the suffix that will be appended to an entity type name when the **Entity Name Mode** has been set to either **guess** or **suffix**.
  
  • With the **Unbound Number as Double** switch, specify whether decimal columns and parameters with no precision or scale should be automatically mapped as Double.
    
    • If set to **OFF** (default), decimal columns and parameters with no precision or scale are not automatically mapped as Double.
    
    • If set to **ON**, decimal columns and parameters with no precision or scale are automatically mapped as Double.

11. Click the **Review Schema Map** tab to review the OData schema map in JSON format.
12. Click **Save Map** to save your configuration of the OData schema map.

13. Set OData options to the desired values.

- **Data Source Caching** controls caching of results. A value of 0 results in a stateless session, requiring Hybrid Data Pipeline to access the data store for each request. A value of 1, the default, allows Hybrid Data Pipeline to cache results, improving performance for repeated requests for the same entity. Hybrid Data Pipeline clears the cache after ten minutes of inactivity or at the end of a session.

- **Page Size** controls the number of results returned in one response. By default, the value in this field is 0 which causes Hybrid Data Pipeline to return up to 2,000 top-level entities per response. If the response contains more than 2,000 entities, the first 2,000 entities are returned and the end of the response contains a link that the OData client can use to fetch the next set. You can set the page size by using values from 1 to 10,000. Client requests can also specify the size of results with query parameters.

- **Refresh Result** determines whether Hybrid Data Pipeline returns results from the cache (for entities in the cache) or queries the data source again. A value of 1, the default, allows Hybrid Data Pipeline to satisfy requests from cached results. A value of 0 forces queries to the backend data store. If caching is not enabled, this parameter has no effect.

- **Inline Count Mode** controls how Hybrid Data Pipeline handles requests that include the `$inlinecount` parameter with a value of `allpages`. The response includes the total number of entities that satisfy the query. A value of 0 causes Hybrid Data Pipeline to skip counting. A value of 1 causes Hybrid Data Pipeline to run a separate query to get the count before the query that returns the entities. This can result in the first page of results being returned faster for large result sets for some data store types. A value of 2, the default, causes Hybrid Data Pipeline to fetch all results and calculate the total number before returning the first page of results to the client.

- **Top Mode** allows Hybrid Data Pipeline to better handle requests that include the `$top` parameter. A value of 0, the default, indicates that clients using `$top` to limit result set size will rarely attempt to get additional entities using the `$skip` parameter. A value of 1 indicates that clients generally use `$top` and `$skip` together to paginate results.

- **OData Read Only** controls read/write access. For a new data source definition, this option is not selected by default. For a data source definition where OData was enabled before this option was available, it will be checked by default. Remove the check mark to enable write access.

14. Click **Update** to save your work.
What to do next:

Test your OData-enabled data source as described in Testing data source configurations (OData Version 4) on page 872.

After you create an OData-enabled data source, you can view the status of the schema map generation on the Data Sources screen. The icon besides the OData-enabled data source indicates the status of the schema map generation. The following table provides details of the icons.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Icon" /></td>
<td>The synchronization of the schema map is in progress. The number denotes the percentage of synchronization completed.</td>
</tr>
<tr>
<td><img src="image2.png" alt="Icon" /></td>
<td>The schema map was synchronized successfully.</td>
</tr>
<tr>
<td><img src="image3.png" alt="Icon" /></td>
<td>The schema map was synchronized successfully, but there are some table/column warnings. Hybrid Data Pipeline allows users to know the details of the tables/columns and/or functions that were dropped while generating the OData Model for a given schema map of a DataSource. The number of warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the Schema API on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td><img src="image4.png" alt="Icon" /></td>
<td>Errors occurred while synchronizing the schema map. You must address the errors and synchronize the schema map again. Hybrid Data Pipeline allows users to know the details of the tables and/or columns that were dropped while generating the OData Model for a given schema map of a DataSource. The number of errors/warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the Schema API on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td><img src="image5.png" alt="Icon" /></td>
<td>You must synchronize the schema map again.</td>
</tr>
</tbody>
</table>

Creating a data source group

A data source group contains references to multiple OData-enabled data source definitions, enabling you to access them all with the same resource path.

Take the following steps to create a data source group.

1. In the left navigation panel, click Data Sources to open the Data Sources view.
2. Click the Data Source Groups tab. The Data Source Groups page opens.
3. Click +NEW GROUP. The Create a Data Source Group page opens.
4. Enter a unique name to identify the data source group in the Data Source Group Name field.
5. Optionally, enter a description for the data source group in the **Description** field.

6. Choose the OData version for the data source group from the **OData Version** dropdown. If you want to use more than one version, you need to create different data source groups for each OData version. Note that the OData version of a data source group cannot be different from the OData versions of its members.

7. Optionally, specify a value in the **Maximum Length of Entity Name** field to control the length of the entity prefix. You can specify values from 10 to 128. Names that are longer than the specified value are altered to fit.

   **OData Access URI** displays the URI to access the data source group. You cannot edit this field.

   The OData base URL is needed to configure your application to use the OData service for a data source. The base URL for an OData enabled data source is shown in the Access URI field of the OData tab. In an OData-enabled application, select HTTP Basic authentication (user ID and password), and provide your Hybrid Data Pipeline user ID and password. With the base URL and Hybrid Data Pipeline credentials configured, OData queries can be executed on the OData service.

8. Optionally, specify whether the OData service temporarily caches information about the data source. Set the value to 1 to enable caching, and provide better performance for production. Set the value to 0 to disable caching; use 0 when you are configuring the data source.

   Caching the back end connection improves performance when multiple OData queries are submitted to the same data source because the connection does not need to be established for every query.

   When you configure a data source for OData, it is recommended that the OData session caching be disabled. Because changes to the Hybrid Data Pipeline data source connection parameters are not implemented during caching because the connection is established using the old data source definition.

9. The OData Data Source section displays the list of data sources that have been enabled for OData. These data sources have a defined schema map and an associated model.

   a) Select the data sources that you want to add to the group.

   b) For every selected data source, enter a unique prefix. The prefix can be a combination of alphanumeric characters but must not start with a number. The length of prefix must be less than half the value of **Maximum Entity Name Length**. For example, if the value of **Maximum Length of Entity Name** is 10, the prefix must be no more than 5 characters long.

10. Click **Save & Close**. The new data source group is displayed on the Data Source Groups page.

11. Optionally, click the OData URI icon to view the OData URI to access the data source group.

---

**Editing a data source group**

Take the following steps to edit a data source group.

1. In the left navigation panel, click **Data Sources** to open the **Data Sources** view.

2. Click the **Data Source Groups** tab. The **Data Source Groups** page opens.

3. Click the name of the data source group you want to edit. Alternatively, select the check box beside the name of the group, and then click the **Edit** icon. The **Edit Data Source Group** page opens.

4. Make changes to the following settings as desired.

   - Modify values in any of these fields: **Data Source Name**, **Description**, or **Maximum Length of Entity Name**.
   - Specify a different OData version with the **OData Version** dropdown.
Note: The OData version of a data source group cannot be different from the OData versions of its members. Switching to a different version of OData means that all the data sources in the group will be removed from the group. If you want to use more than one version of OData, you must create different data source groups for each version.

- Add new data sources to the data source group.
  1. Under OData Data Sources, click All. A list of all the OData-enabled data sources appears.
  2. Select the data sources you want to add to the group.
- To remove data sources from the data source group, clear the check boxes beside the data sources.

5. Click Save & Close.

Deleting a data source group

Take the following steps to delete a data source group.

Note: Deleting a data source group does not delete the member data sources of the group.

1. In the left navigation panel, click Data Sources to open the Data Sources view.
2. Click the Data Source Groups tab. The Data Source Groups page opens.
3. Select the data source groups you want to delete, and then click Delete.
4. A message to confirm deletion appears. Click Delete.

   The selected data source groups are deleted and removed from the Data Source Name list.

Creating and using REST data sources

Hybrid Data Pipeline supports SQL read-only access to JSON-based REST services through the Autonomous REST Connector. When you create a REST data source, the connector creates a relational model of the returned JSON data and translates SQL statements to REST API requests. You can create and manage REST data sources either through the Web UI or through the Hybrid Data Pipeline API.

See the following topics for more information on creating and using REST data sources.

- Creating REST data sources through the Web UI
  - How to create a data source in the Web UI on page 225. When creating a data source to connect to a REST service, select the Autonomous REST Connector data store.
  - Autonomous REST Connector parameters on page 257. Once you select the Autonomous REST Connector data store, specify values for parameters to define the REST data source.
- Creating REST data sources with the API on page 638
- Creating an input REST file on page 641
Creating REST data sources with the API

The following operations should be performed to set up and review a REST data source using Hybrid Data Pipeline APIs.

- Create a REST data source
- Upload an input REST file
- Test the REST data source
- Retrieve the input REST file
- Retrieve the output REST file

Create a REST data source

Use the Data Sources API to create a REST data source. The following example creates a REST data source called TestREST. Values for the name, dataStore, connectionType, and options parameters must be specified. All REST data sources are created by way of the Autonomous REST Connector data store with an ID of 62 and with a connection type of Cloud.

Request

POST https://MyServer:8443/api/mgmt/datasources

Request Payload

```json
{
  "name": "TestREST",
  "dataStore": 62,
  "connectionType": "Cloud",
  "description": "Test REST ds definition",
  "options": {
    "User": "test",
    "Password": "test",
    "ODataVersion": "4",
    "AuthenticationMethod": "Basic"
  }
}
```

Response Payload

```json
{
  "id": 956,
  "name": "TestREST",
  "dataStore": 62,
  "connectionType": "Cloud",
  "description": "Test REST ds definition",
  "options": {
    "User": "test",
    "Password": "test",
    "ODataVersion": "4",
    "AuthenticationMethod": "Basic"
  }
}
```
Upload an input REST file

Use the Driver Files API to upload an input REST file. REST endpoints must be provided either via the Web UI or by uploading an input REST file must be uploaded. As shown in the following example, the request includes the data source ID that was generated when the data source was created (956). In the request payload, the input REST file is provided in the form of a JSON object. (See Creating an input REST file on page 641 for syntax requirements.)

Request

```
POST https://MyServer:8443/api/mgmt/datasources/956/export/driverfiles/inputrest
```

Request Payload

```
{
  "countries": {
    "#path": "http://example.com/country",
    "#get": {
      "start_date": "2018-08-31",
      "end_date": "2018-09-01",
      "departments": "[engineering,marketing,sales]",
      "tags": "[blue,green,red]"
    }
  }
}
```

Response Payload

```
{
  "countries": {
    "#path": "http://example.com/country",
    "#get": {
      "start_date": "2018-08-31",
      "end_date": "2018-09-01",
      "departments": "[engineering,marketing,sales]",
      "tags": "[blue,green,red]"
    }
  }
}
```

Test the REST data source

Use the Data Sources API to test REST connectivity. In the following example, values for user and password are specified to allow for basic authentication with the REST service.

Request

```
POST https://MyServer:8443/api/mgmt/datasources/956/test
```

Request Payload

```
{
  "user": "test",
  "password": "test"
}
```

Response Payload

```
{
  "success": true
}
```
Retrieval the input REST file
The Driver Files API can be used to retrieve the input REST file for review.

Request

GET https://MyServer:8443/api/mgmt/datasources/956/export/driverfiles/inputrest

Response Payload

```
{
  "countries": {
    "#path": "http://example.com/country",
    "#get": {
      "start_date": "2018-08-31",
      "end_date": "2018-09-01",
      "departments": "[engineering,marketing,sales]",
      "tags": "[blue,green,red]"
    }
  }
}
```

Retrieval the output REST file
The output REST file is created at the time of the test connection. The output REST file is a JSON file that maps the relational view of the REST endpoints provided in the input REST file. A review of the output REST file may be useful for developing an input REST file and creating better SQL queries to run against a REST service.

Request

GET https://MyServer:8443/api/mgmt/datasources/956/export/driverfiles/outputrest

Response Payload

```
{
  "countries": {
    "#path": ["https://example.com/country"],
    "type": "VarChar(64),#key",
    "metadata": {
      "generated": "BigInt",
      "url": "VarChar(184)",
      "title": "VarChar(64)",
      "status": "Integer",
      ...
    },
    "features[1]": {
      "type": "VarChar(10)",
      "properties": {
        "size": "Decimal",
        "place": "VarChar(108)",
        ...
      },
      "geometry": {
        "type": "VarChar(7)",
        "coordinates[3]": "Double"
      },
      "id": "VarChar(27)"
    },
    ...
  }
}
```
Creating an input REST file

The input REST file is a JSON file which specifies one or more REST endpoints in the form of a JSON object. The input REST file may include only endpoints, or it can include endpoints with parameters that define the REST data. When initially connecting to a REST endpoint, Hybrid Data Pipeline uses the input REST file to build a relational model of the REST data. You can create an input REST file with a text editor. Once you create the input REST file, it can be uploaded via the Web UI or with the Drive Files API.

The basic format of the input REST file consists of a list of comma-separated endpoints. The following example shows how endpoints are mapped as tables to support a relational schema.

```
{
  "<table_name1>"":"<endpoint1>",
  "<table_name2>"":"<endpoint2>",
  "<table_name3>"":"<endpoint3>"
}
```

**Note:** The syntax requirements described here can also be applied to editing the relational model of your REST data through the Web UI. It should also be noted that the Entity Name field in the Web UI specifies the name of the relational table.

Valid formats for the input REST file are described in detail in the following sections.

- Specifying Endpoints for GET Request with Unparameterized Paths
- Specifying Endpoints for GET Request with Parameterized Paths
- Specifying Endpoints for GET Requests with Query Parameters
- Specifying Endpoints for Requests with Custom HTTP Headers
- Defining a POST Request
- Configuring Paging

**Specifying Endpoints for GET Request with Unparameterized Paths**

To specify endpoints for unparameterized GET requests, use the following format:

```
"<table_name>"":"<host_name>/<endpoint_path>"
```

- **table_name**
  
  is the name of the relational table to which the driver maps the endpoint. For example, country.

- **host_name**
  
  (optional) is the protocol and host name components of the URL endpoint. For example, http://example.com. You can omit this value by specifying the host name using the ServerName property.
Specifying Endpoints for GET Request with Parameterized Paths

To specify parameterized GET requests, use the following format:

"<table_name>"":"<host_name>/<endpoint_path1>/{<param_name>:<param_value>}[/<endpoint_path2>]"

**table_name**

is the name of the relational table to which the driver maps the endpoint. For example, states.

**host_name**

(optional) is the protocol and host name components of the URL endpoint. For example,.
http://example.com. You can omit this value by specifying the host name using the ServerName property.

**endpoint_path**

is the path component of the URL endpoint. For example, states.

**param_name**

is the parameter identifier used for filtering the request. For example, countryCode.

**param_value**

is the parameter value used for filtering the request during sampling. For example, USA.

For example, the following demonstrates a GET request that will map to the states table.

"states":"http://example.com/states/get/{countryCode:USA}/all"

Specifying Endpoints for GET Requests with Query Parameters

Use the following format to specify endpoints for GET requests with argument parameters. Multiple argument parameters withing the same endpoint are separated by an ampersand (&).

"<table_name>"":"<host_name>/<endpoint_path>?<parameter>=<value>[&...]"

**table_name**

is the name of the relational table to which the driver maps the endpoint. For example, timeseries.

**host_name**

(optional) is the protocol and host name components of the URL endpoint. For example,.
http://example.com. You can omit this value by specifying the host name using the ServerName property.
endpoint_path

is the path component of the URL endpoint. For example, times.

parameter

is the argument parameter component of the parameter=value pair used for filtering the request. For example, interval.

value

is the value argument parameter used for filtering the request. For example, 5min.

For example, the following demonstrates a GET request that will map to the timeseries table.

"timeseries": "https://www.example.com/times/query?interval=5min&symbol=USA&function=TIME_SERIES_WEEKLY"

Specifying Endpoints for Requests with Custom HTTP Headers

Some endpoints employ custom HTTP headers to filter data returned by a GET request. This type of filtering is typically used to create multiple unique reports/tables from the same endpoint. To use custom headers, you must define the request in the input REST file. The REST file entry is comprised of a path and header object. The path object contains the URL endpoint used in requests, while the header object defines the headers and provides value arguments used to filter the request.

In addition to filtering requests, the header object can be used to specify a value for the Accept header if the default, application/json, is not accepted by the endpoint. This scenario typically occurs when accessing a vendor endpoint that uses a proprietary Accept header.

An entry for a GET request using custom HTTP headers takes the following form:

"table_name": {
    "#path": "<host_name>/<endpoint_path>",
    "#headers": {
        "<header1>": "<value1>",
        "<header2>": "<value2>",
        "<header3>": "<value3>"
    }
}

table_name

is the name of the relational table to which the driver maps the endpoint. For example, people.

host_name

(optional) is the protocol and host name components of the URL endpoint. For example, http://example.com. You can omit this value by specifying the host name using the ServerName property.

endpoint_path

is the path component of the URL endpoint. For example, times.

header

is the HTTP header component of the header=value pair used for filtering the request. For example, X-Subway-Payment.

When overriding the Accept header, this value is Accept.
The value argument for the HTTP header used for filtering the request or, if overriding the default Accept header, the value of the Accept header for the endpoint. For example, token.

For example, the following demonstrates an entry for a GET request that defines custom HTTP headers.

```
"people": {
    "#path": "http://example.com/people",
    "#headers": {
        "Accept": "application/calendar+json",
        "X-Subway-Payment": "token",
        "X-Laundry-Service": "dryclean",
        "X-Favorite-Food": "pizza"
    }
}
```

### Defining a POST Request

To use POST requests, you must define the request in the REST file in the JSON format. The definition entry is comprised of a path and body. The path contains the URL endpoint and the body used in requests, while the body defines documents and provides sample values. The driver then uses these sample values to define which data type to be used when executing a POST request. An entry for a POST request takes the following form:

```
"table_name": {
    "#path": "<host_name>/<endpoint_path>",
    "#post": {
        "<field1>": "<value1>",
        "<field2>": "<value2>",
    }
}
```

- **table_name**
  
  The name of the relational table to which the driver maps the endpoint. For example, countries2.

- **host_name**
  
  (optional) is the protocol and host name components of the URL endpoint. For example, http://example.com. You can omit this value by specifying the host name using the ServerName property.

- **endpoint_path**
  
  is the path component of the URL endpoint. For example, country.

- **document**
  
  is the document name of the document=value pair. For example, START_DATE.

- **value**
  
  is the sample value the driver uses to determine the data type to use when executing a POST to that document. For example, 2018-08-31.

For example, the following demonstrates an entry for a POST request that will map to the countries2 table.

```
"countries2": {
    "#path": "http://example.com/country/",
```
Configuring Paging

The driver supports two types of paging: offset and page numbering paging. To configure paging, specify values for the properties in the following tables that correspond to the type of paging you want to employ. Paging properties can be set for individual GET or POST requests by specifying these options in the body object. If paging properties are not specified, the driver will attempt to retrieve the first page for data sources that require paging.

The following demonstrates configuring row offset paging for an unparametrized GET request:

```
"table_name": {
   "#path": "<host_name>/<endpoint_path>",
   "#maximumPageSize":1000,
   "#firstRowNumber":1,
   "#pageSizeParameter":"maxResults",
   "#rowOffsetParameter":"startAt"
}
```

table_name

is the name of the relational table to which the driver maps the endpoint. For example, countries2.

host_name

(optional) is the protocol and host name components of the URL endpoint. For example, http://example.com. You can omit this value by specifying the host name using the ServerName property.

data source

is the path component of the URL endpoint. For example, country.

### Table 126: Row Offset Paging Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#maximumPageSize</td>
<td>The maximum page size in rows.</td>
</tr>
<tr>
<td>#firstRowNumber</td>
<td>The number of the first row. The default is 0; however, some systems begin numbering rows at 1.</td>
</tr>
<tr>
<td>#pageSizeParameter</td>
<td>The name of the URI parameter that contains the page size.</td>
</tr>
<tr>
<td>#rowOffsetParameter</td>
<td>The name of the URI parameter that contains the starting row number for this set of rows.</td>
</tr>
</tbody>
</table>
Table 127: Page Number Paging Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#maximumPageSize</td>
<td>The maximum page size in rows.</td>
</tr>
<tr>
<td>#firstPageNumber</td>
<td>The number of the first page. The default is 0; however, some systems begin numbering pages at 1.</td>
</tr>
<tr>
<td>#pageSizeParameter</td>
<td>The name of the URI parameter that contains the page size.</td>
</tr>
<tr>
<td>#pageNumberParameter</td>
<td>When requesting a page of rows, this is the name of the URI parameter to contain the page number.</td>
</tr>
</tbody>
</table>

See also
Sample Input REST File on page 646

Sample Input REST File

The following is a sample input REST file that demonstrates GET requests, POST requests, and a request configured for paging.

```json
{
    // a simple GET request without parameters to sample:
    "countries": "http://example.com/country",

    // A GET request with a parameter in the path:
    "states": "http://example.com/states/get/{countryCode:USA}/all",

    // A GET request with parameters as arguments
    "timeseries": "https://www.example.com/times/query?interval=5min&symbol=USA&function=TIME_WEEKLY",

    // A GET request with parameters as arguments
    "timeseries": "https://www.example.com/times/query?interval=5min&symbol=USA&function=TIME_WEEKLY",

    // A GET request with custom HTTP headers
    "people": {
        "#path": "http://example.com/people",
        "#headers": {
            "Accept": "application/calendar+json",
            "X-Subway-Payment": "token",
            "X-Laundry-Service": "dryclean",
            "X-Favorite-Food": "pizza"
        }
    },

    // A POST with parameters in the body
    "countries2": {
        "#path": "http://example.com/country",
        "#post": {
            "start_date": "2018-08-31",
            "end_date": "2018-09-01",
            "departments": ["engineering,marketing,sales"],
            "tags": ["blue,green,red"]
        }
    }
}
```
// A GET with paging configured
"products": {  
  "#path": "http://example.com/products",
  "#maximumPageSize": 1000,
  "#firstRowNumber": 1,
  "#pageSizeParameter": "maxResults",
  "#rowOffsetParameter": "startAt"
}
Configuring Hybrid Data Pipeline Driver for ODBC

For details, see the following topics:

• Getting started with the ODBC Driver
• Supported features
• Configuring an ODBC data source on UNIX and Linux systems
• Configuring and testing an ODBC data source on Windows
• Connecting applications to the connectivity service
• Connection properties reference
• Application considerations
• Troubleshooting
• Internationalization, localization, and Unicode
• Code page values
• WorkAround options
Getting started with the ODBC Driver

Progress DataDirect Hybrid Data Pipeline Driver for ODBC works with the Hybrid Data Pipeline connectivity service to provide SQL access to supported cloud data stores from any ODBC-compliant application.

Information that the driver needs to connect to a database is stored in a data source. The ODBC specification describes three types of data sources: user data sources, system data sources (not a valid type on UNIX/Linux), and file data sources. On Windows, user and system data sources are stored in the registry of the local computer. The difference is that only a specific user can access user data sources, whereas any user of the machine can access system data sources. On all platforms, file data sources, which are simply text files, can be stored locally or on a network computer, and are accessible to other machines.

When you define and configure a data source, you store default connection values for the driver that are used each time you connect to a particular database. You can change these defaults by modifying the data source. For information on installing the driver, refer to the Progress DataDirect Hybrid Data Pipeline Installation Guide.

See Configuring an ODBC data source on UNIX and Linux systems on page 651 for information on setting environment variables, configuring a data source in the system information file, and setting up DSN-less connections.

See Configuring and testing an ODBC data source on Windows on page 658 for information on defining a data source in the ODBC Administrator.

Application considerations on page 700 provides information on verifying the driver version number, retrieving data type information, and supported ODBC API functions and scalar functions.

Troubleshooting on page 708 provides information on identifying where an issue originates, creating a trace log, and using ODBC Test.

Supported features

This section describes how the Hybrid Data Pipeline Driver for ODBC implements standard ODBC, security, and connectivity features.

Data encryption

All communication between the driver and the connectivity service, including user IDs and passwords, is encrypted using Secure Sockets Layer (SSL). SSL is an industry-standard protocol for sending encrypted data over connections. It secures the integrity of your data by encrypting information and providing client/server authentication. In addition, you have the option of storing the credentials for your cloud data store securely in the cloud data source, or of managing it yourself in the ODBC data source.

Unicode

Multilingual ODBC applications can be developed on any operating system using the driver to access both Unicode and non-Unicode enabled data stores. The driver is fully Unicode enabled. On UNIX and Linux platforms, the driver supports both UTF-8 and UTF-16.
On Windows platforms, the driver supports UCS-2/UTF-16 only. The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

The driver supports the Unicode ODBC W (Wide) function calls, such as SQLConnectW. This allows the Driver Manager to transmit these calls directly to the driver. Otherwise, the Driver Manager would incur the additional overhead of converting the W calls to ANSI function calls, and vice versa.

See Internationalization, localization, and Unicode on page 715 for related details.

Safe thread handling

The ODBC specification mandates that all drivers must be thread-safe, that is, drivers must not fail when database requests are made on separate threads. The ODBC 3.0 specification does not provide a method to find out how a driver services threaded requests, although this information is useful to an application.

The Hybrid Data Pipeline for ODBC driver provides this information to the user through the SQLGetInfo information type 1028, returning a value of 1. A return value of 1 denotes that the session is restricted at the connection level, that is, one thread per connect.

Sessions of this type are fully thread-enabled when simultaneous threaded requests are made with statement handles that do not share the same connection handle. In this model, if multiple requests are made from the same connection, the first request received by the driver is processed immediately and all subsequent requests are serialized.

Number of connections and statements supported

The driver supports multiple connections and multiple statements per connection.

Parameter metadata

The driver supports returning parameter metadata for all types of SQL statements and stored procedure arguments.

SQL support

The Hybrid Data Pipeline Driver for ODBC, working in conjunction with the Hybrid Data Pipeline connectivity service, supports standard SQL 92. Specific support is determined by the data store to which the Hybrid Data Pipeline connectivity service is connected. For example, the SQL supported by Salesforce is different than the SQL supported by Oracle.

Configuring an ODBC data source on UNIX and Linux systems

The Hybrid Data Pipeline Driver for ODBC is supported on UNIX and Linux systems. Before using the driver on a UNIX or Linux system, an ODBC data source must be configured to work with the driver.
Chapter 4: Configuring Hybrid Data Pipeline Driver for ODBC

The following procedures require that you have the appropriate permissions to modify your environment and to read, write, and execute various files. You must log in as a user with full r/w/x permissions set recursively across the entire ODBC driver installation directory.

Take the following steps to configure an ODBC data source and test a connection to a Hybrid Data Pipeline data source using the ODBC driver.

1. Check your permissions. You must log in as a user with full r/w/x permissions that apply recursively across the entire ODBC driver installation directory.

2. Set environment variables by running the appropriate product setup script.

   **Note:** Alternatively, you can set environment variables manually. See Setting environment variables manually on page 653 for details.

   a) Determine which shell you are running by executing `echo $SHELL` from your login shell.

   b) Run the appropriate product setup script.

      • For Bourne, Korn, and related shells, execute the following command:

        ```
        . ./odbc.sh
        ```

      • For C shell and related shells, execute the following command:

        ```
        source ./odbc.csh
        ```

   c) Execute `env` to verify that the following environment variables have been set accordingly.

      • Library search path environment variable. The name of the library search path environment variable depends on the platform you are using.

        • `LD_LIBRARY_PATH=/<install_dir>/lib` (HP-UX IPF, AIX 5.2 and later, Linux, and Oracle Solaris)
        • `LIBPATH=/<install_dir>/lib` (AIX 5.1 and earlier)
        • `SHLIB_PATH=/<install_dir>/lib` (HP-UX PA-RISC)

        • `ODBCINI=/<install_dir>/odbc.ini`
        • `ODBCINST=/<install_dir>/odbcinst.ini`
        • `ODBCINST` is required for DSN-less connections on page 657.

3. Edit the system information file as described in Configuring a data source in the system information file on page 655.

4. Test the connection to your data source as described in Example application for UNIX and Linux on page 658.

**See also**

Setting environment variables manually on page 653
Setting environment variables manually

Instead of using the product setup scripts installed with the ODBC driver, you can manually set environment variables needed to configure an ODBC data source and use the driver. The following topics provide instruction on manually setting the relevant environment variables.

Before proceeding, you must have the appropriate permissions to modify your environment and to read, write, and execute various files. You must log in as a user with full r/w/x permissions set recursively across the entire ODBC driver installation directory.

Library Search Path

The library search path environment variable must be set so that the driver and ODBC core components can be located at the time of execution.

Library search path environment variable. The name of the library search path environment variable depends on the platform you are using.

- **LD_LIBRARY_PATH** on HP-UX IPF, AIX 5.2 and later, Linux, and Oracle Solaris
- **LIBPATH** on AIX 5.1 and earlier
- **SHLIB_PATH** on HP-UX PA-RISC

In the following examples, **LD_LIBRARY_PATH** is being set to point to the location of shared libraries for an installation on `/opt/odbc`.

In the C shell, you would set this variable as follows:

```
setenv LD_LIBRARY_PATH /opt/odbc/lib
```

In the Bourne or Korn shell, you would set it as:

```
LD_LIBRARY_PATH=/opt/odbc/lib;export LD_LIBRARY_PATH
```

To verify that the **LD_LIBRARY_PATH** environment variable has been set, execute the `env` command and review the output to confirm.

ODBCINI

The product installation directory includes a default system information file, named **odbc.ini**. This file can be renamed or moved to another location. In either case, the environment variable **ODBCINI** must be set to point to the fully qualified path name of the .ini file that you want to use for data source configuration.

For example, to point to the location of the file for an installation on `/opt/odbc` in the C shell, you would set this variable as follows:

```
setenv ODBCINI /opt/odbc/odbc.ini
```

In the Bourne or Korn shell, you would set it as:

```
ODBCINI=/opt/odbc/odbc.ini;export ODBCINI
```

To verify that the **ODBCINI** environment variable has been set, execute the `env` command and review the output to confirm.

As an alternative, you can choose to make the **odbc.ini** file a hidden file and not set the **ODBCINI** variable. In this case, you would need to rename the file to `.odbc.ini` (to make it a hidden file) and move it to the user’s `~$HOME` directory.

The driver searches for the location of the **odbc.ini** file as follows:
1. The driver checks the ODBCINI variable.
2. The driver checks $HOME for .odbc.ini.

If the driver does not locate the system information file, it returns an error.

**See also**

Configuring a data source in the system information file on page 655

**ODBCINST**

The product installation directory includes a default file named odbcinst.ini for use with DSN-less connections. This file can be renamed or moved to another location. In either case, the environment variable ODBCINST must be set to point to the fully qualified path name of the .ini file.

For example, to point to the location of the file for an installation on /opt/odbc in the C shell, you would set this variable as follows:

```
setenv ODBCINST /opt/odbc/odbcinst.ini
```

In the Bourne or Korn shell, you would set it as:

```
ODBCINST=/opt/odbc/odbcinst.ini;export ODBCINST
```

To verify that the ODBCINST environment variable has been set, execute the `env` command and review the output to confirm.

As an alternative, you can choose to make the odbcinst.ini file a hidden file and not set the ODBCINST variable. In this case, you would need to rename the file to .odbcinst.ini (to make it a hidden file) and move it to the user's $HOME directory.

The driver searches for the location of the odbcinst.ini file as follows:

1. The driver checks the ODBCINST variable.
2. The driver checks $HOME for .odbcinst.ini.

If the driver does not locate the odbcinst.ini file, it returns an error.

**See also**

DSN-less connections on page 657

**DD_INSTALLDIR**

The DD_INSTALLDIR environment variable provides the driver with the location of the product installation directory so that it can access support files. If the InstallDir property has not been set in your .ini file(s), then DD_INSTALLDIR must be set to point to the fully qualified path name of the installation directory.

DD_INSTALLDIR overrides the InstallDir setting in any .ini files that are in use.

For example, to point to the location of the directory for an installation on /opt/odbc in the C shell, you would set this variable as follows:

```
setenv DD_INSTALLDIR /opt/odbc
```

In the Bourne or Korn shell, you would set it as:

```
DD_INSTALLDIR=/opt/odbc;export DD_INSTALLDIR
```

To verify that the DD_INSTALLDIR environment variable has been set, execute the `env` command and review the output to confirm.
The driver searches for the location of the installation directory as follows:

1. The driver checks the **DD_INSTALLDIR** variable.
2. The driver checks the **odbc.ini** or the **odbcinst.ini** files for the **InstallDir** keyword.

If the driver does not locate the installation directory, it returns an error.

### Configuring a data source in the system information file

To configure a data source in UNIX and Linux environments, you must edit the system information file to which the **ODBCINI** variable points.

You can use the **odbc.ini** file installed with the driver as a template for the system information file. Using a text editor, modify the default attributes in this file as necessary, based on your system values (for example, your server name and port number).

To use Hybrid Data Pipeline with an ODBC application, you need to configure an ODBC data source that connects to a Hybrid Data Pipeline data source. The following table describes how the entries in the ODBC data source map to a Hybrid Data Pipeline data source.

**Table 128: ODBC parameters for connecting to a Hybrid Data Pipeline data source**

<table>
<thead>
<tr>
<th>ODBC data source parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HybridDataPipelineDataSource</strong></td>
<td>The name of the Hybrid Data Pipeline data source to which the ODBC data source will connect.</td>
</tr>
<tr>
<td><strong>LogonID</strong></td>
<td>The user name for your Hybrid Data Pipeline account.</td>
</tr>
<tr>
<td><strong>ODBC Data Source</strong></td>
<td>A unique name for the ODBC data source. Specified in the [ODBC Data Sources] section of the system file, for example, DataDirect HDP=MyHDPDataSource.</td>
</tr>
<tr>
<td><strong>DataSourcePassword</strong></td>
<td>If the credentials of a database or data store (such as Oracle Database or Salesforce) are not stored in the Hybrid Data Pipeline data source, provide the database or data store password.</td>
</tr>
<tr>
<td><strong>DataSourceUser</strong></td>
<td>If the credentials of a database or data store (such as Oracle Database or Salesforce) are not stored in the Hybrid Data Pipeline data source, provide the database or data store user name.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The password for your Hybrid Data Pipeline account.</td>
</tr>
<tr>
<td><strong>EncryptionMethod</strong></td>
<td>The method the driver uses to encrypt data sent between the driver and the Hybrid Data Pipeline server.</td>
</tr>
<tr>
<td><strong>PortNumber</strong></td>
<td>The port number on which the Hybrid Data Pipeline service is listening. The default value is 8080.</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>The DNS name of the machine where Hybrid Data Pipeline is installed.</td>
</tr>
</tbody>
</table>
See [Connecting applications to the connectivity service](#) on page 659 for information on how to configure your application to use an ODBC data source.

**Sample odbc.ini file**

You can use the odbc.ini file installed with the driver as a template for the system information file. The following sample shows how this file can be modified. All occurrences of ODBCHOME should be replaced with your installation directory path during installation of the file. Values that you must supply are enclosed by angle brackets (< >). If you are using the installed odbc.ini file, you must supply the values and remove the angle brackets.

**Note:** The prefix for the 32-bit driver file name is iv. The prefix for the 64-bit driver file name is dd.

```plaintext
[ODBC Data Sources]
DataDirect HDP=<odbc_data_source_name>

[ODBC]
IANAAppCodePage=4
InstallDir=ODBCHOME
Trace=0
TraceFile=odbctrace.out
TraceDll=ODBCHOME/lib/ddtrc27.so

[DataDirect HDP]
Driver=ODBCHOME/lib/ddhybrid01.so
Description=
Service=<hdp_dns_name>
HybridDataPipelineDataSource=<hdp_data_source>
LogonID=<hdp_user_name>
ClientTimeZone=
DataSourceUser=<datastore_user_name>
DataSourcePassword=<datastore_user_password>
ProxyHost=
ProxyPort=
ProxyUser=
ProxyPassword=
TransactionMode=0
WSRetryCount=3
WSTimeout=120
LogonDomain=
LoginTimeout=30
QueryTimeout=0
ApplicationUsingThreads=1
ReportCodepageConversionErrors=0
EnableWCharSupport=1
DefaultLongDataBufLen=
MaxVarcharSize=
ExtendedOptions=
VarcharThreshold=
MinLongVarcharSize=
Password=<hdp_user_password>
EncryptionMethod=0
ValidateServerCertificate=1
TrustStore=
TrustStorePassword=
HostNameInCertificate=
PortNumber=80
```
DSN-less connections

Connections to a data source can be made via a connection string without referring to a data source name (DSN-less connections). This is done by specifying the DRIVER= keyword instead of the DSN= keyword in a connection string, as outlined in the ODBC specification. A file named odbcinst.ini must exist when the driver encounters DRIVER= in a connection string such as the following:

```
Driver=DataDirect HDP 4.2;HybridDataPipelineDS=MyPipelineDS;DataSourcePassword=myDSpw;
DataSourceUser=John.johnson@company.com;LogonID=John;Password=myPipelinepw;
Service=my.pipeline.host.name;PortNumber=8080;EncryptionMethod=0;
```

The ODBC driver installation program installs a default version of the odbcinst.ini file in the product installation directory. This is a plain text file that contains default DSN-less connection information. You should not normally need to edit this file.

The content of this file may include a section named [ODBC]. The [ODBC] section in the odbcinst.ini file fulfills the same purpose in DSN-less connections as the [ODBC] section in the odbc.ini file does for data source connections. If the information in these two sections is not the same, the values in the odbc.ini [ODBC] section override those of the odbcinst.ini [ODBC] section.

See also
ODBCINST on page 654
ODBCINI on page 653
Configuring a data source in the system information file on page 655

Sample odbcinst.ini file

The following is a sample odbcinst.ini file. All occurrences of ODBCHOME should be replaced with your installation directory path during installation of the file. Commented lines are denoted by the # symbol.

```
[ODBC Drivers]
DataDirect HDP 4.2=Installed

[DataDirect Hybrid Data Pipeline Driver]
APILevel=1
ConnectFunctions=YYY
Driver=ODBCHOME/lib/ddhybrid01.so
DriverODBCVer=3.52
FileExtns=.*
FileUsage=1
HelpRootDirectory=ODBCHOME/help
Setup=ODBCHOME/lib/ddhybrid01.so
SQLLevel=1

[ODBC]
#This section must contain values for DSN-less connections
#if no odbc.ini file exists. If an odbc.ini file exists, 
#the values from that [ODBC] section are used.
IANAAppCodePage=4
InstallDir=ODBCHOME
Trace=0
```

Note: The prefix for the 32-bit driver file name is .iv. The prefix for the 64-bit driver file name is dd.
Example application for UNIX and Linux

Progress DataDirect ships an application, named example, that is installed in the /samples/example subdirectory of the product installation directory. Once you have configured your environment and data source, use the example application to test passing SQL statements. To run the application, enter example and follow the prompts to enter your data source name, user name, and password.

If successful, a SQL> prompt appears and you can type in SQL statements, such as SELECT * FROM table_name. If example is unable to connect to the database, an appropriate error message appears.

Refer to the example.txt file in the example subdirectory for an explanation of how to build and use this application.

Configuring and testing an ODBC data source on Windows

On Windows systems, you can configure and modify data sources through the ODBC Data Source Administrator, which is available from the Hybrid Data Pipeline Driver for ODBC program group. You specify default connection values in the driver’s setup dialog box. The ODBC Data Source Administrator stores the values as user or system data sources in the Windows Registry, or as file data sources in a specified location. See “Creating a Data Source definition in the User’s Guide.”

The following steps describe how to configure and test a connection to a Hybrid Data Pipeline connectivity service Data Source as a user data source. You must create a Hybrid Data Pipeline Data Source with the dashboard before using these procedures.

1. From the Progress Hybrid Data Pipeline Driver for ODBC program group in your start menu, start the ODBC Data Source Administrator.
2. On the User DSN tab, click Add to add a new data source.

You can add a file or system data source using similar procedures.

3. Scroll down the list of drivers and select Progress DataDirect Hybrid Data Pipeline.
4. Click Finish.

The Progress DataDirect Hybrid Data Pipeline ODBC Driver window displays with focus on the General tab.

5. For Data Source Name, enter a unique name for this data source, such as myHybridDSN.
6. Optionally enter a description in the Description field.
7. For Hybrid Data Pipeline Source, enter the name of the Hybrid Data Pipeline Data Source you created in the Hybrid Data Pipeline dashboard, such as MyForceDS.
8. For Service, type the DNS name of the machine where the Hybrid Data Pipeline service is installed.
9. For Port, type the port number that the Hybrid Data Pipeline service is listening to.
10. Click Test Connection and a dialog box will prompt your for credentials.
11. Enter the user name and password for your Hybrid Data Pipeline user account.
Note: You can optionally store the Hybrid Data Pipeline account credentials in the ODBC Data Source by entering them on the Security tab.

12. If you did not store the cloud data store credentials in the cloud Data Source, select the More Options check box and enter the user name and password for the cloud data store. If the account requires a security token, append it to the password.

13. Click OK.

A dialog box is displayed to inform you whether the connection was successful.

14. Configure your ODBC-compliant application to use this data source.

See Example application for Windows on page 659 to use the installed example application to test SQL queries.
See Connecting applications to the connectivity service on page 659 for information on how to configure your application to use this data source.

Example application for Windows

Progress DataDirect ships an application, named EXAMPLE.EXE, that is installed in the \samples\example subdirectory of the product installation directory. Once you have configured your environment and data source, use the example application to test passing SQL statements.

Refer to the EXAMPLE.TXT file in the example subdirectory for an explanation of how to build and use this application.

Connecting applications to the connectivity service

Packaged applications such as SAP Crystal Reports, Microsoft Access or Excel and custom applications can specify the information to connect to the Hybrid Data Pipeline connectivity service using an ODBC data source and/or passing connection properties in an ODBC connection string.

The connection parameters set in the ODBC data source or passed in a connection string from the application only apply to the connection between the application and Hybrid Data Pipeline Driver for ODBC. The Hybrid Data Pipeline Data Source that you create configures the connection to the data store, such as Salesforce.com or Oracle Service Cloud.

Note: Before modifying an application, check the requirements for library compatibility with Hybrid Data Pipeline Driver for ODBC in the Installation Guide.

Connection strings

The example in Connecting applications to the connectivity service on page 659 shows how to create an ODBC data source definition. Applications can use such a named data source definition to connect to the Hybrid Data Pipeline connectivity service. You also have the option of providing the required connection properties through the application instead of saving them in an external location. And you can use a combination of a data source and connection properties, for example, to avoid storing credentials in the data source.

ODBC keywords to specify connection information in an application include:
Chapter 4: Configuring Hybrid Data Pipeline Driver for ODBC

- **DSN** specifies a named data source:
  
  ```
  DSN=data_source_name[;attribute=value[;attribute=value]...]
  ```

- **FILEDSN** specifies a filename, where the file contains the data source information

  ```
  FILEDSN=filename.dsn[;attribute=value[;attribute=value]...]
  ```

- **DRIVER** includes the connection parameters in the connection string and should supply all required properties:

  ```
  DRIVER=[[]driver_name[]][;attribute=value[;attribute=value]...]
  ```

All of these keywords allow `attribute=value` pairs in the connection string. You can use these to specify connection properties that customize behavior. For the **DSN** and **FILEDSN**, keywords, the values specified in the connection string override the ODBC data source values for connection properties.

The following examples show how to override the user name and password on Windows systems for an ODBC data source named Pipeline:

- DSN=Pipeline;UID=test@abccorp.com;PWD=XYZZY
- FILEDSN=Pipeline;UID=test@abccorp.com;PWD=XYZZY

**Connection Properties** on page 661 describes required and optional connection properties.

### File data sources

A file data source is simply a text file that contains connection information. It can be created with a text editor. The file normally has an extension of `.dsn`. The advantage of a file data source is that it can be stored on a server and accessed by other machines, either Windows, UNIX, or Linux.

The Driver Manager on UNIX and Linux supports file data sources. On Windows systems, you can use the ODBC Administrator to create a file data source. See Getting started with the ODBC Driver on page 650 for a general description of ODBC data sources on both Windows and UNIX.

The file data source is accessed by specifying the **FILEDSN** instead of the **DSN** keyword in a connection string, as outlined in the ODBC specification. The complete path to the file data source can be specified in the syntax that is normal for the machine on which the file is located. For example, on Windows:

- FILEDSN=C:\Program Files\Common Files\ODBC\DataSources\Hybridwp.dsn

  or, on UNIX and Linux:

  - FILEDSN=/home/users/john/filedsn/Hybridwp2.dsn

If no path is specified for the file data source, the Driver Manager uses the DefaultDSNDir property, which is defined in the [ODBC File DSN] setting in the odbc.ini file to locate file data sources. If the [ODBC File DSN] setting is not defined, the Driver Manager uses the InstallDir setting in the [ODBC] section of the odbc.ini file. The Driver Manager does not support the SQLReadFileDSN and SQLWriteFileDSN functions.

As with any connection string, you can specify attributes to override the default values in the data source:

- FILEDSN=/home/users/john/filedsn/Hybrid.dsn;UID=john;PWD=test01
A file data source for the Hybrid Data Pipeline driver would be similar to the following:

```plaintext
[ODBC]
Driver=DataDirect Hybrid Data Pipeline 4.1
LogonID=JOHN
HybridDataPipelineDataSource=SALES
LoginTimeout=15
```

It must contain all basic connection information plus any optional attributes. Because it uses the DRIVER keyword, an `odbcinst.ini` file containing the driver location must exist (see Configuring an ODBC data source on UNIX and Linux systems on page 651).

### Connection Properties

Regardless of the method an application uses to connect to the Hybrid Data Pipeline connectivity service, certain properties are required to connect. Optional properties control behavior of the communication between the Hybrid Data Pipeline Driver for ODBC and the Hybrid Data Pipeline connectivity service.

Connection properties can be supplied in a Data Source or `.ini` file or passed in a connection string using the `Driver` keyword. You can split them and specify some in a Data Source or `.ini` file and pass others in a connection string. The latter is common to protect credentials for security reasons.

### Required properties

The following table lists the properties required to connect. If you are using the ODBC Administrator to add or modify a data source (DSN), you will find them on the **General** and **Security** tabs of the **Progress DataDirect Hybrid Data Pipeline ODBC Driver** dialog box.

**Table 129: Required Connection Properties**

<table>
<thead>
<tr>
<th>Required</th>
<th>Property Name</th>
<th>Field Location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required</td>
<td>User Name</td>
<td></td>
<td>User name for your account.</td>
</tr>
<tr>
<td></td>
<td>LogonID (UID):Security tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required</td>
<td>Password</td>
<td></td>
<td>Password for your account. You can specify this in a connection string or the user will be prompted for this value.</td>
</tr>
<tr>
<td></td>
<td>Password (PWD):Logon to Hybrid Data Pipeline dialog box--→</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required</td>
<td>Hybrid Data Pipeline Source</td>
<td></td>
<td>Name of the Hybrid Data Pipeline Data Source defined in <code>http://&lt;myserver&gt;:8080/d2c-ui</code>, where <code>myserver</code> is the DNS name of the machine where Hybrid Data Pipeline is installed.</td>
</tr>
<tr>
<td></td>
<td>HybridDataPipelineDataSource (HDPDS):General tab</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Required</td>
<td>Service</td>
<td></td>
<td>The DNS name of the machine where Hybrid Data Pipeline is installed.</td>
</tr>
<tr>
<td></td>
<td>Service (SRVC):General tab</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Required

<table>
<thead>
<tr>
<th>Required if the Hybrid Data Pipeline <strong>Data Source</strong> does not contain credentials for the data store.</th>
<th><strong>Property Name</strong></th>
<th><strong>Field Location</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Data Source User</strong></td>
<td>DataSourceUser (DSU): <strong>Security</strong> tab</td>
<td>Account user name for the data store if it is not provided in the Hybrid Data Pipeline <strong>Data Source</strong> or in the connection string.</td>
</tr>
<tr>
<td></td>
<td><strong>Data Source Password</strong></td>
<td>DataSourcePassword (DSP): <strong>Security</strong> tab</td>
<td>Account password for the cloud data store if it is not provided in the Hybrid Data Pipeline <strong>Data Source</strong> or in the connection string.</td>
</tr>
<tr>
<td>Required if the value of the server port has been changed.</td>
<td><strong>Port Number</strong></td>
<td>PortNumber</td>
<td>The port number on which the Hybrid Data Pipeline service is listening. The default value is specified during the installation of the Hybrid Data Pipeline server.</td>
</tr>
</tbody>
</table>

### General Tab

The **General** tab displays fields that are required for creating a data source. The fields on all other tabs are optional, unless noted otherwise.
A connection string using the `DRIVER` keyword must provide all necessary connection information:

```
Driver={Progress DataDirect Hybrid DataPipeline 4.1};HybridDataPipelineDataSource=My_DataSource;LoginTimeout=100;LogonID=HDP_Login;Password=HDP_Password
```

If an application does not provide the credentials required to connect to Hybrid Data Pipeline, depending on how the application is implemented, the user can receive an error or a Logon dialog box provided by the connectivity service.

**Logon to Hybrid Data Pipeline**

The following screen shot shows the Logon to Hybrid Data Pipeline dialog box. The values a user must enter correspond to the values shown in Required properties on page 661.
Data Source configuration in the UNIX/Linux odbc.ini File

On UNIX and Linux, you must set up the proper ODBC environment before configuring data sources. Data sources for UNIX and Linux are stored in the system information file (by default, odbc.ini).

You can configure and modify data sources directly by editing the odbc.ini file and storing default connection values there. See Configuring a data source in the system information file on page 655 for detailed information about the specific steps necessary to configure a data source.

Connection properties reference on page 676 lists driver connection string attributes that must be used in the odbc.ini file to set the value of the attributes. Note that only the long name of the attribute can be used in the file. The default listed in the table is the initial default value when the driver is installed.

Optional Connection Properties

Hybrid Data Pipeline Driver for ODBC has initial default values for some connection properties, making it optional for you to set them. On Windows systems, the ODBC Hybrid Driver Setup dialog box displays these values when you create a data source. On UNIX and Linux systems, the ODBC.ini file created by the installer contains the connection properties that you can define.

You can change connection property values in the following ways:

• By modifying them in a data source using the ODBC Administrator, in the Windows Registry, or by editing an odbc.ini file
• By overriding them in DSN or FILEDSN connection strings
• By specifying them in a DRIVER connection string

Many connection properties also have short names for use in connection strings as a convenience. For a full description of each property, or to look them up alphabetically, see Connection properties reference on page 676.

The connection properties are organized by functionality on the tabs of the Progress DataDirect Hybrid Data Pipeline ODBC Driver dialog setup box.

• Advanced functionality
• Security features
• Web Service configuration features
• Proxy server configuration features
See the tables in the topic for each tab for the property names that you can use in .ini files, the Windows Registry, or connection strings, the default value and a brief description.

**Advanced Tab Options**

The **Advanced** tab contains the following fields:

The following table describes fields in the **Advanced** tab of the Progress DataDirect Hybrid Data Pipeline ODBC Driver setup dialog box, lists the initial default values, and provides the long and short name of the corresponding property. The long name of properties can be set in the Windows registry or in a .ini file, as described in Configuring a Data Source in the System Information File. The short name of properties can be passed in connection strings.
### Table 130: Advanced Tab Options

<table>
<thead>
<tr>
<th>Field Name (Short name)</th>
<th>Description</th>
<th>Initial Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application Using Threads</strong></td>
<td>If selected or set to 1, the ODBC driver works with single-threaded and multi-threaded applications. A value of 0 disables multi-threading. For more information, see Application Using Threads on page 678.</td>
<td>Enabled</td>
</tr>
<tr>
<td><strong>Login Timeout</strong></td>
<td>The number of seconds the ODBC driver waits for a connection to be established before returning control to the application and generating a timeout error. A value of -1 or 0 prevents timeouts. For more information, see Login Timeout on page 686.</td>
<td>0</td>
</tr>
<tr>
<td><strong>Query Timeout</strong></td>
<td>The number of seconds before timeout for all statements that are created by a connection. A value of 0 prevents a query from timing out. For more information, see Query Timeout on page 692.</td>
<td>0</td>
</tr>
<tr>
<td><strong>Report Codepage Conversion Errors</strong></td>
<td>Determines what will happen if a character cannot be converted from one character set to another, allowed values are:</td>
<td>0 - Ignore Errors</td>
</tr>
<tr>
<td></td>
<td>• If set to 0 - Ignore Errors, the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• If set to 1 - Return Error, the driver returns an error instead of substituting 0x1A for unconverted characters.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• If set to 2 - Return Warning, the driver substitutes 0x1A for each character that cannot be converted and returns a warning.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see Report Codepage Conversion Errors on page 693.</td>
<td></td>
</tr>
<tr>
<td>Field Name (Short name)</td>
<td>Description</td>
<td>Initial Default Value</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>Transaction Mode</strong></td>
<td>If set to 0 - No Transactions, the data source and the driver do not support transactions. Metadata indicates that the driver does not support transactions. If set to 1 - Ignore, the data source does not support transactions and the driver always operates in auto-commit mode. Calls to set the driver to manual commit mode and to commit transactions are ignored. Calls to rollback a transaction cause the driver to return an error indicating that no transaction is started. Metadata indicates that the driver supports transactions and the ReadUncommitted transaction isolation level. If set to 2 - Transactions, the data source and driver support manual transactions for supported data stores. Support for isolation levels depends on which backend data store is being used. If the data store does not support transactions (for example, Salesforce), then Transaction Mode is switched to 0 - No Transactions. See also Transaction Mode on page 694.</td>
<td>2 - Transactions</td>
</tr>
</tbody>
</table>
| **Client Time Zone**   | Specifies a time zone for time and timestamp values that will be applied by the data store. The driver by default attempts to determine the timezone of the client. If it can not determine that timezone automatically, specify the client time zone to use by setting a value for ClientTimezone. The format is:  

<timezone>,<+ or ->HH:MM<D>  

where D specifies to account for daylight savings time. For example:  

America/New_York,-05:00D  

or  

America/New_York,-5D  

For more information see Client Time Zone on page 679. | Empty (the driver uses the client time zone, based on the system-specific time zone settings) |
<table>
<thead>
<tr>
<th>Field Name (Short name)</th>
<th>Description</th>
<th>Initial Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enable WChar Support</strong></td>
<td>Specifies whether the driver maps character data to the the ODBC Unicode data types, such as WCHAR, WVARCHAR, or WLONGVARCHAR. When using an application that does not support Unicode data types, disable this option. The driver then maps character data to an ANSI Char type, such as CHAR, VARCHAR, or LONGVARCHAR.</td>
<td>Enabled</td>
</tr>
<tr>
<td>EnableWCharSupport(EWS)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Default Buffer Size for Long/LOB Columns (in Kb)</strong></td>
<td>Specifies the maximum length of data (in Kb) the driver can send using the SQL_DATA_AT_EXEC parameter.</td>
<td>1024</td>
</tr>
<tr>
<td>DefaultLongDataBufLen(DLBL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Max Varchar Size</strong></td>
<td>Specifies the maximum size of columns of type SQL_VARCHAR that the driver describes through result set descriptions and catalog functions. Allowed value is a positive integer from 1 to ( x ) where ( x ) is the maximum size of the SQL_VARCHAR data type. If you leave the field empty, the actual size of the columns from the database is persisted to the application.</td>
<td>empty</td>
</tr>
<tr>
<td>MaxVarcharSize(MVS)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field Name (Short name)</td>
<td>Description</td>
<td>Initial Default Value</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>VarChar Threshold</strong></td>
<td>Specifies the threshold at which the driver describes columns of the data type SQL_VARCHAR as SQL_LONGVARCHAR. If the size of the SQL_VARCHAR column exceeds the value specified, the driver describe the column as SQL_LONGVARCHAR when calling SQLDescribeCol and SQLColumns. This option allows you to fetch columns that would otherwise exceed the upper limit of the SQL_VARCHAR type for some third-party applications. Allowed value is a positive integer from 1 to ( x ) where ( x ) is the maximum size in characters of columns the driver will describe as SQL_VARCHAR.</td>
<td>empty</td>
</tr>
<tr>
<td><strong>VarCharThreshold (VT)</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| **Min Long Varchar Size** | Specifies the minimum count of characters the driver reports for columns mapped as SQL_LONGVARCHAR. If the size of a SQL_LONGVARCHAR column is less than the value specified, the driver increases the reported size of the column to this value when calling SQLDescribeCol and SQLColumns. This allows you to fetch SQL_LONGVARCHAR columns whose size is smaller than the minimum imposed by some third-party applications, such as SQL Server Linked Server. Allowed value is a positive integer from 1 to \( x \) where \( x \) is the minimum size in characters the driver reports for columns mapped to the SQL_LONGVARCHAR type. | empty |
| **MinLongVarcharSize (MINLVS)** | | |

**Extended Options**: Type a semi-colon separated list of connection options and their values. Use this configuration option to set the value of undocumented connection options that are provided by Progress DataDirect technical support. You can also add the WorkAround connection option in the Extended Options string, for example:

```
Workaround=9;Option1=value [];Option2=value;
```

If the Extended Options string contains option values that are also set in the setup dialog or data source, the values of the options specified in the Extended Options string take precedence. However, connection options that are specified on a connection string override any option value specified in the Extended Options string.

Optionally, click the **Security** tab to specify security data source settings.

**See also**

Using the WorkAround Options on page 728
Security Tab Options

The following table describes fields in the Security tab of the Progress DataDirect Hybrid Data Pipeline ODBC Driver Setup dialog box, and provides the long name of the corresponding property that you can set in a .ini file and the short name for connection string attributes. The fields on this tab are required, but can be supplied in different ways as described in Required properties on page 661.

The Security tab contains the following fields, which have no initial default value:

![Security tab screenshot]

Table 131: Security Tab Options

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Property Name (Short name)</th>
<th>Description</th>
<th>Initial Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td></td>
<td>Specifies the username. For more information, see User Name on page 696.</td>
<td>None</td>
</tr>
<tr>
<td>LogonID (UID)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field Name Property Name (Short name)</td>
<td>Description</td>
<td>Initial Default Value</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------</td>
<td>-----------------------</td>
<td></td>
</tr>
<tr>
<td>Logon Domain LogonDomain (LD)</td>
<td>Specifies the domain part of the Hybrid Data Pipeline connectivity service user id for applications that do not handle an @ character. If Logon Domain is not an empty string, the driver first appends the @ character to the end of the User Name value and then appends the value of Logon Domain, allowing use of an e-mail address as a user name. For more information, see Logon Domain on page 687.</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Data Source User DataSourceUser (DSU)</td>
<td>Account user name for the data store if it is not provided in the Data Source or connection string. For example, if a Hybrid Data Pipeline Data Source is configured to connect to Salesforce, the value for Data Source User is your Salesforce User ID. For more information, see Data Source Name on page 680.</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Data Source Password DataSourcePassword (DSP)</td>
<td>Account password for the cloud data store if they are not provided in the Data Source or connection string. For more information, see Data Source Password on page 680.</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Enable SSL EncryptionMethod (EM)</td>
<td>The method the driver uses to encrypt data sent between the driver and the Hybrid Data Pipeline server. If not enabled (the default), data is not encrypted. If selected, the driver uses an SSL protocol.</td>
<td>Disabled (the check box is not selected, or the value is set to 0 in the connection string)</td>
<td></td>
</tr>
<tr>
<td>Validate Server Certificate ValidateServerCertificate (VSC)</td>
<td>Determines whether the connectivity service validates the certificate that is sent by the Hybrid Data Pipeline server when SSL encryption is enabled. If set to 0 (Disabled) or false, the connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any trust store information specified by the Trust Store and Trust Store Password options. If set to 1 (Enabled) or true, the connectivity service validates the certificate that is sent by the database server.</td>
<td>Enabled (the check box is selected)</td>
<td></td>
</tr>
<tr>
<td>Trust Store TrustStore (TS)</td>
<td>Specifies the location of the trust store file that contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. An absolute path is recommended.</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Field Name</td>
<td>Property Name (Short name)</td>
<td>Description</td>
<td>Initial Default Value</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Trust Store Password</td>
<td>TrustStorePassword (TSP)</td>
<td>The password that is used to access the trust store file when server authentication is used. The trust store file contains a list of the Certificate Authorities (CAs) that the client trusts.</td>
<td>None</td>
</tr>
<tr>
<td>Host Name In Certificate</td>
<td>HostNameInCertificate (HNIC)</td>
<td>Specifies a host name or server name that is validated against the information stored in an SSL certificate when validation is enabled (ValidateServerCertificate=1).</td>
<td>None</td>
</tr>
</tbody>
</table>

**Web Service Tab options**

The following table describes fields in the Web Service tab of the Progress DataDirect Hybrid Data Pipeline ODBC Driver Setup dialog box and provides the long name of the corresponding property that you can set in a .ini file and a short name for connection string attributes. These settings apply to communication between the Hybrid Data Pipeline Driver for ODBC and the Hybrid Data Pipeline connectivity service. The communication between the Hybrid Data Pipeline connectivity service and the data store is configured in the Data Source.
The **Web Service** tab contains the following fields:
### Table 132: Web Service Tab Options

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Property Name (Short name)</th>
<th>Description</th>
<th>Initial Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WSRetry Count</strong></td>
<td>WSRetryCount (WSRC)</td>
<td>The number of times the driver retries a timed-out Select request. Insert, Update, and Delete requests are never retried. The timeout period is specified by the WSTimeout connection option. For more information, see WSRetryCount on page 698.</td>
<td>3</td>
</tr>
<tr>
<td><strong>WSTimeout</strong></td>
<td>WSTimeout (WST)</td>
<td>Specifies the time, in seconds, that the driver waits for a response to a Web service request. For more information, see WSTimeout on page 699.</td>
<td>120</td>
</tr>
</tbody>
</table>

### Proxy tab options

If you need to connect to the Hybrid Data Pipeline connectivity service through a proxy server that requires authentication, provide values for the fields on this tab. The following table describes fields in the **Proxy** tab of the Progress DataDirect Hybrid Data Pipeline ODBC Driver Setup dialog box and provides the long and short name of the corresponding property that you can set in a .ini file or using connection string attributes. These settings apply to communication between the Hybrid Data Pipeline Driver for ODBC and the Hybrid Data Pipeline connectivity service. They have no initial default values.
### Table 133: Proxy Tab Options

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Property Name (Short name)</th>
<th>Description</th>
<th>Initial Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy Host</td>
<td>ProxyHost (PXHN)</td>
<td>The Hostname and possibly the Domain of the Proxy Server. The value specified can be a host name, a fully qualified domain name, or an IPv4 or IPv6 address. For more information, see Proxy Host on page 690.</td>
<td>None</td>
</tr>
<tr>
<td>Proxy Port</td>
<td>ProxyPort (PXPT)</td>
<td>The port number where the Proxy Server is listening for HTTP and/or HTTPS requests. For more information, see Proxy Port on page 691.</td>
<td>None</td>
</tr>
</tbody>
</table>
Connecting Through a Proxy Server

In some environments, your application may need to connect through a proxy server. At a minimum, your application needs to provide the following connection information if the application connects through a proxy server:

- Server name or IP address of the proxy server
- Port number on which the proxy server is listening for HTTPS requests

In addition, if authentication is required, your application may need to provide a valid user ID and password for the proxy server. Consult with your system administrator for the required information.

If your environment requires a proxy server, the connection information for the proxy server can be specified in the `ProxyHost`, `ProxyPort`, `ProxyUser`, and `ProxyPassword` connection attributes. See Proxy tab options on page 674 for details about these attributes.

Connection properties reference

The connection properties in this section are listed alphabetically by the name that appears on the driver setup dialog box or the logon dialog box. The attribute name and short name, which can be used in connection strings, data source files, and `.ini` file data source sections are listed underneath the GUI name.

**Note:** The connection properties described in this section configure the connection between the application (through the Hybrid Data Pipeline Driver for ODBC) and the Hybrid Data Pipeline connectivity service. The data source defined on the Hybrid Data Pipeline dashboard configures the connection between the Hybrid Data Pipeline connectivity service and the Hybrid Data Pipeline data store. See Connecting applications to the connectivity service on page 659 for more information.

In most cases, the GUI name and the property name are the same; however, some exceptions exist. Also, a few connection string attributes do not have equivalent GUI options. They are listed alphabetically by their attribute names.
# ODBC Connection Properties

The following table lists connection properties alphabetically, with links to the appropriate description, and lists the default values.

**Table 134: ODBC Connection Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Using Threads</td>
<td>1 (Enabled)</td>
</tr>
<tr>
<td>Client Time Zone</td>
<td>Empty string (use system time zone)</td>
</tr>
<tr>
<td>Data Source Name</td>
<td>None</td>
</tr>
<tr>
<td>Data Source Password</td>
<td>None</td>
</tr>
<tr>
<td>Data Source User</td>
<td>None</td>
</tr>
<tr>
<td>Default Buffer Size for Long/LOB Columns (in Kb) on page 682</td>
<td>1024</td>
</tr>
<tr>
<td>Description</td>
<td>None</td>
</tr>
<tr>
<td>Enable SSL</td>
<td>0 - Disabled</td>
</tr>
<tr>
<td>Enable WChar Support</td>
<td>1 - Enabled</td>
</tr>
<tr>
<td>Host Name In Certificate on page 684</td>
<td>None</td>
</tr>
<tr>
<td>Hybrid Data Pipeline Source</td>
<td>None</td>
</tr>
<tr>
<td>IANAAppCodePage</td>
<td>4 (ISO 8559-1 Latin-1)</td>
</tr>
<tr>
<td>Login Timeout</td>
<td>0</td>
</tr>
<tr>
<td>Logon Domain</td>
<td>None</td>
</tr>
<tr>
<td>Max Varchar Size</td>
<td>None</td>
</tr>
<tr>
<td>Min Long Varchar Size</td>
<td>None</td>
</tr>
<tr>
<td>Password</td>
<td>None</td>
</tr>
<tr>
<td>Port Number on page 690</td>
<td>None</td>
</tr>
<tr>
<td>Proxy Host</td>
<td>Empty string</td>
</tr>
<tr>
<td>Proxy Password</td>
<td>Empty string</td>
</tr>
<tr>
<td>Proxy User on page 692</td>
<td>Empty string</td>
</tr>
<tr>
<td>Proxy User</td>
<td>Empty string</td>
</tr>
</tbody>
</table>
### Application Using Threads

**Attribute**

ApplicationUsingThreads (AUT)

**Purpose**

Determines whether the driver works with multi-threaded ODBC applications. This connection option can affect performance.

**Valid Values**

0 | 1

**Behavior**

If set to 1 (Enabled), the driver works with single-threaded and multi-threaded applications.

If set to 0 (Disabled), the driver does not work with multi-threaded applications. If using the driver with single-threaded applications, this value avoids additional processing required for ODBC thread-safety standards.

**Default**

1 (Enabled)

**GUI Tab**

Advanced tab
**Client Time Zone**

**Attribute**

ClientTimeZone (CTZ)

**Purpose**

Specifies the time zone other than UTC (Universal Time Coordinated) that should be used when translating data store time and timestamp values between the Hybrid Data Pipeline service and the driver. Most data stores use UTC, while applications use and supply time and timestamp values in local time for the client. If the ClientTimeZone value is not specified (the initial default value), the driver uses system-specific client time zone settings to convert between the UTC time used by the cloud service and the local time used by the client application. The driver returns an error at connect time if it cannot obtain the client time zone. Client time zone settings vary among operating systems:

- On Windows systems, the driver translates the system time zone to an equivalent client time zone.
- On UNIX and Linux systems, the client time zone is set using the $TZ$ variable, with the following exceptions:
  - On Linux systems, if $TZ$ is NULL or empty, the client time zone comes from the ZONE value in the following file: /etc/sysconfig/clock
  - On Solaris, $TZ$ may contain localtime. In this case, the driver uses the /etc/localtime link to determine the client time zone.

If you want the connectivity service to verify that this time zone has certain characteristics, you can append some verification information onto the time zone, such as an offset and daylight saving time indicator. The offset specification is the number of hours (and optional minutes) behind (indicated by a leading minus sign) or ahead (indicated by no sign or a plus sign) of UTC. If the time zone is expected to support daylight saving time, append D to the offset.

**Valid Values**

timezone[,+[ | -]HH[:MM]][D]

where:

timezone

is a valid Java TimeZone ID. See your Java documentation or use the TimeZone.getAvailableIDs() method to return a list of valid IDs.

+ | -

optionally specifies whether the offset is before or after Greenwich Mean Time.

HH:MM

optionally specifies the number of hours and minutes to offset the time from Greenwich Mean Time.

D

signifies whether the time zone adjusts for daylight savings time.
Example
America/New_York
America/New_York,-5D
America/New_York,-05:00D
Asia/Calcutta,5:30
Asia/Calcutta,+5:30

Default
Empty (the driver determines the client time zone based on the system-specific time zone settings)

GUI Tab
Advanced tab

Data Source Name

Attribute
DataSourceName (DSN)

Purpose
Specifies a unique name for an ODBC data source configuration.

Valid Values
string
where:
string
is the name of a data source.

Example
Accounting or Pipeline to Salesforce Data

Default
None

GUI Tab
General tab

Data Source Password

Attribute
DataSourcePassword (DSP)
**Purpose**
Specifies the case-sensitive password that is required for logging into a backend data store, such as SQL Server or Salesforce. For web service data stores such as Salesforce, a security token may be required by the data store instance.

**Valid Values**

```
password | password+securitytoken
```

where:

```
password
```

is the password required for logging into the data store.

```
password+securitytoken
```

is the password required for logging into the data store plus a valid security token.

**Notes**

- The data store user ID and password may be stored in the Hybrid Data Pipeline data source definition. If that is true and you specify the user ID and password using the `DataSourceUser` and `DataSourcePassword` connection attributes, the values specified in these connection attributes take precedence.

- When the data store requires a security token but it has not been stored in the Hybrid Data Pipeline data source definition, you must append the security token to the end of the password specified for `DataSourcePassword`. In the example `secretXaBARTsLZReM4Px47qPLOS`, `secret` is the password and the remainder of the value is the security token.

- All communication between the driver and the Hybrid Data Pipeline connectivity service is encrypted using SSL, including the values specified for `DataSourceUser` and `DataSourcePassword`.

**Default**
None

**GUI Tab**
`Security` tab

---

**Data Source User**

**Attribute**
`DataSourceUser (DSU)`

**Purpose**
Specifies the user ID that is required for logging into a backend data store, such as SQL Server or Salesforce.

**Valid Values**

```
string
```

where:
is the user ID required for logging into the data store.

**Notes**

- The data store user ID and password may be stored in the Hybrid Data Pipeline data source definition. If that is true and you specify the user ID and password using the `DataSourceUser` and `DataSourcePassword` connection attributes, the values specified in these connection attributes take precedence.
- All communication between the driver and the Hybrid Data Pipeline connectivity service is encrypted using SSL, including the values specified for `DataSourceUser` and `DataSourcePassword`.

**Default Buffer Size for Long/LOB Columns (in Kb)**

**Attribute**

`DefaultLongDataBufLen` (DBDBL)

**Purpose**

Specifies the maximum length of data (in KB) the driver can fetch from long columns in a single round trip and the maximum length of data that the driver can send using the SQL_DATA_AT_EXEC parameter.

**Valid Values**

Any integer greater than 0

**Default**

1024

**GUI Tab**

`Advanced` tab

**Description**

**Attribute**

`Description` (n/a)

**Purpose**

Specifies an optional long description of a data source. This description is not used as a runtime connection attribute, but does appear in the `ODBC.INI` section of the Registry and in the `odbc.ini` file.
Valid Values

*string*

where:

*string*

is a description of a data source.

Example

My Customer Data

Default

None

GUI Tab

General tab

Enable SSL

Attribute

*EncryptionMethod (EM)*

Purpose

Specifies whether the driver encrypts data sent between the driver and the database server.

Valid Values

0 | 1

Behavior

If set to 0 (not selected), data is not encrypted
If set to 1 (selected), the driver uses TLS1 data encryption.

Default

0 (not selected)

GUI Tab

Advanced tab

Enable WChar Support

Attribute

*EnableWCharSupport (EWS)*
**Purpose**

Specifies whether the driver maps character data to the ODBC Unicode data types, such as WCHAR, WVARCHAR, or WLONGVARCHAR. By default, the driver maps character data to the ODBC Unicode data types, sometimes called W-Types. Hybrid Data Pipeline always returns character data as Unicode, using the UTF-8 character encoding.

Some applications do not support the Unicode data types. When using this type of application, disable the Enable WChar Support option. The driver then maps character data to an ANSI Char type, such as CHAR, VARCHAR, or LONGVARCHAR.

**Valid Values**

1 (Enabled) | 0 (Disabled)

**Behavior**

If set to 1 (Enabled), the driver maps character data to the ODBC W-types, such as WCHAR, WVARCHAR, or WLONGVARCHAR. Character data is returned in Unicode when retrieved as SQL_C_DEFAULT.

If set to 0 (Disabled), the driver maps character data to a Char type, such as CHAR, VARCHAR, or LONGVARCHAR. Character data is returned in IANAAppCodePage.

**Default**

1 (Enabled). The driver maps character data to the ODBC Unicode types.

**GUI Tab**

Advanced tab

**Host Name In Certificate**

**Attribute**

HostNameInCertificate (HNIC)

**Purpose**

A host name that is validated against the information stored in an SSL certificate when validation is enabled (ValidateServerCertificate=1). This option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested. This option is only valid when SSL encryption is enabled.

**Valid values**

host_name | #SERVERNAME#

where:

host_name

is the host name specified in the certificate. Consult your SSL administrator for the correct value.
Behavior

If the value is set to a host name, the driver examines the subjectAltName values included in the certificate. If a dnsName value is present in the subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the dnsName value. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the dnsName value.

If no subjectAltName values exist or a dnsName value is not in the list of subjectAltName values, then the driver compares the value specified for Host Name In Certificate with the commonName part of the Subject name in the certificate. The commonName typically contains the host name of the machine for which the certificate was created. The connection succeeds if the values match. The connection fails if the Host Name In Certificate value does not match the commonName.

If multiple commonName parts exist in the Subject name of the certificate, the connection succeeds if the Host Name In Certificate value matches any of the commonName parts.

Default
None

GUI tab
Security tab

See also
Data encryption

Hybrid Data Pipeline Source

Attribute
HybridDataPipelineDataSource (HDPDS)

Purpose
Specifies the Hybrid Data Pipeline Data Source to use for a connection.

Valid Values
datasource_name
where:
datasource_name
is the name of a valid Hybrid Data Pipeline Data Source defined in the connectivity service.

Default
None

GUI Tab
General tab
IANAAppCodePage

**Attribute**
IANAAppCodePage (IACP)

**Purpose**
An Internet Assigned Numbers Authority (IANA) value. On UNIX and Linux, you must specify a value for this option if your application is not Unicode-enabled or if your database character set is not Unicode. See Code page values on page 722 for details.

The driver uses the specified IANA code page to convert "W" (wide) functions to ANSI.

The driver and Driver Manager both check for the value of IANAAppCodePage in the following order:

- In the connection string
- In the Data Source section of the system information file (odbc.ini)
- In the ODBC section of the system information file (odbc.ini)

If the driver does not find an IANAAppCodePage value, the driver uses the default value of 4 (ISO 8859-1 Latin-1).

**Valid Values**
IANA_code_page

where:

IANA_code_page

is one of the valid values listed in Code page values on page 722. The value must match the database character encoding and the system locale.

**Default**
4 (ISO 8859-1 Latin-1)

Login Timeout

**Attribute**
LoginTimeout (LT)

**Purpose**
Specifies the number of seconds the Hybrid Data Pipeline Driver for ODBC waits for a connection to be established before returning control to the application and generating a timeout error. To override the value that is set by this connection option for an individual connection, set a different value in the SQL_ATTR_LOGIN_TIMEOUT connection attribute using the SQLSetConnectAttr() function.

**Valid Values**
-1 | 0 | x
where:

\( x \)

is a positive integer that represents a number of seconds.

**Behavior**

If set to -1, the connection request does not time out. The driver silently ignores the `SQL_ATTR_LOGIN_TIMEOUT` attribute.

If set to 0, the connection request does not time out, but the driver responds to the `SQL_ATTR_LOGIN_TIMEOUT` attribute.

If set to \( x \), the connection request times out after the specified number of seconds unless the application overrides this setting with the `SQL_ATTR_LOGIN_TIMEOUT` attribute.

**Default**

0

**GUI Tab**

**Advanced** tab

### Logon Domain

**Attribute**

LogonDomain (LD)

**Purpose**

Specifies the domain part of the Hybrid Data Pipeline connectivity service user ID. If Logon Domain is not an empty string, the driver first appends the @ character to the end of the User Name value and then appends the value of Logon Domain.

Some applications do not allow you to configure a username for an ODBC data source that contains an @ character in the name. However, Hybrid Data Pipeline user IDs can be in the form of an email address that contains the @ character. To facilitate these types of applications, the user id can be specified in two parts, the name and the domain.

**Example**

To specify the user name of `john.doe@mycompany.com` to an application that does not allow an @ character in the login name, set the User ID to John.Doe and the Login domain to mycompany.com. If a value is specified for Login Domain, the driver appends the @ character to the end of the user name, and then appends the login domain after that.

**Valid Values**

`string`

where:

`string`

is a valid user ID domain.
Max Varchar Size

Attribute
MaxVarcharSize (MVS)

Purpose
Specifies the maximum size of columns of type SQL_VARCHAR that the driver describes through result set descriptions and catalog functions.

Valid Values
A positive integer from 1 to \( x \)
where:

\[ x \]

is the maximum size of the SQL_VARCHAR data type.

Default
None. The actual size of the columns from the database is persisted to the application.

Min Long Varchar Size

Attribute
MinLongVarcharSize (MINLVS)

Purpose
Specifies the minimum count of characters the driver reports for columns mapped as SQL_LONGVARCHAR. If the size of a SQL_LONGVARCHAR column is less than the value specified, the driver will increase the reported size of the column to this value when calling SQLDescribeCol and SQLColumns. This allows you to fetch SQL_LONGVARCHAR columns whose size is smaller than the minimum imposed by some third-party applications, such as SQL Server Linked Server.

Valid values
A positive integer from 1 to \( x \)
where:
is the minimum size in characters the driver will report for columns mapped to the SQL_LONGVARCHAR type.

Notes

• Configuring the Varchar Threshold and Min Long Varchar Size options allows you to fetch SQL_VARCHAR and SQL_LONGVARCHAR columns with sizes that fall between the data-type ranges used by some applications.

Default

None. If no value is specified, the driver does not change the column size reported for SQL_LONGVARCHAR columns.

GUI tab

Advanced tab

Password

Attribute

Password (PWD)

Purpose

Specifies the password to use to connect to the Hybrid Data Pipeline connectivity service. A password is required.

Important: Setting the password using an ODBC data source is not recommended. The ODBC data source persists all options, including passwords, in clear text. Set the password through the Logon dialog box or a connection string.

Valid Values

pwd

where

pwd

is a valid password for the specified Hybrid Data Pipeline connectivity service account. The password is case-sensitive.

Default

None

GUI Tab

N/A
Port Number

Attribute

PortNumber (PORT)

Purpose

Specifies the port number that the Hybrid Data Pipeline service is listening to for HTTP or HTTPS requests. The default value is specified during the installation of the Hybrid Data Pipeline server.

Valid Values

port_name

where:

port_name

is the port number of the Hybrid Data Pipeline service listener.

Default

None

GUI Tab

Advanced tab

Proxy Host

Attribute

ProxyHost (PXHN)

Purpose

Specifies the Hostname and possibly the Domain of the Proxy Server. The value specified can be a host name, a fully qualified domain name, or an IPv4 or IPv6 address.

Valid Values

server_name | IP_address

where:

server_name

is the name of the server or a fully qualified domain name to which you want to connect. Check with your system administrator for the correct server name or IP address.

Default

Empty string
GUI Tab
Proxy tab

Proxy Password

Attribute
ProxyPassword (PXPW)

Purpose
Specifies the password needed to connect to the Proxy Server, if a password is required.

Valid Values
String
where:

String

  specifies the password to use to connect to the Proxy Server. Contact your system administrator to obtain your password.

Default
Empty string

GUI Tab
Proxy tab

Proxy Port

Attribute
ProxyPort (PXPT)

Purpose
Specifies the port number where the Proxy Server is listening for HTTPS requests.

Valid Values
port_name
where:

port_name

  is the port number of the server listener. Check with your system administrator for the correct number.

Default
0
GUI Tab
Proxy tab

Proxy User

Attribute
ProxyUser (PXUN)

Purpose
Specifies the user name needed to connect to the Proxy Server.

Valid Values
The default user ID that is used to connect to the Proxy Server. Contact your system administrator to obtain your user ID for the proxy server, if a user ID is required.

Default
Empty string

GUI Tab
Proxy tab

Query Timeout

Attribute
QueryTimeout (QT)

Purpose
Specifies the number of seconds for the default query timeout for all statements that are created by a connection. To override the value set by this connection option for an individual statement, set a different value in the SQL_ATTR_QUERY_TIMEOUT statement attribute on the SQLSetStmtAttr() function.

Valid Values
-1 | 0 | x
where:

x

is a number of seconds.

Behavior
If set to -1, the query does not time out. The driver silently ignores the SQL_ATTR_QUERY_TIMEOUT attribute.
If set to 0, the query does not time out, but the driver responds to the SQL_ATTR_QUERY_TIMEOUT attribute.
If set to \( x \), all queries time out after the specified number of seconds unless the application overrides this value by setting the `SQL_ATTR_QUERY_TIMEOUT` attribute.

**Default**

0

**GUI Tab**

**Advanced** tab

### Report Codepage Conversion Errors

**Attribute**

`ReportCodepageConversionErrors (RCCE)`

**Purpose**

Specifies how the driver handles code page conversion errors that occur when a character cannot be converted from one character set to another.

An error message or warning can occur if an ODBC call causes a conversion error, or if an error occurs during code page conversions to and from the database or to and from the application. The error or warning generated is `Code page conversion error encountered`. In the case of parameter data conversion errors, the driver adds the following sentence: `Error in parameter x, where x is the parameter number`. The standard rules for returning specific row and column errors for bulk operations apply.

**Valid Values**

0 | 1 | 2

**Behavior**

If set to 0 (Ignore Errors), the driver substitutes 0x1A for each character that cannot be converted and does not return a warning or error.

If set to 1 (Return Error), the driver returns an error instead of substituting 0x1A for unconverted characters.

If set to 2 (Return Warning), the driver substitutes 0x1A for each character that cannot be converted and returns a warning.

**Default**

0 (Ignore Errors)

**GUI Tab**

**Advanced** tab

### Service

**Attribute**

`Service (SRVC)`
**Purpose**
Specifies the connectivity service to which the driver connects.

**Valid Values**
\(<myserver>:<port>\)
where \(<myserver>\) is the DNS name or the IP address of the machine where Hybrid Data Pipeline is installed.

*Note:* Unless the ports 80 and 443 are redirected to 8080 and 8443 respectively, you must specify \(<myserver>:<port>\).

**Behavior**
The driver connects to the Hybrid Data Pipeline connectivity service.

**Default**
\(<myserver>:<port>\)
where \(<myserver>\) is the DNS name or the IP address of the machine where Hybrid Data Pipeline is installed.

**GUI Tab**
General tab

**Transaction Mode**

**Attribute**
TransactionMode (TM)

**Purpose**
Specifies how the driver handles manual transactions.

**Valid Values**
0 1 2

**Behavior**

If set to 0 - No Transactions, the data source and the driver do not support transactions. Metadata indicates that the driver does not support transactions.

If set to 1 - Ignore, the data source does not support transactions and the driver always operates in auto-commit mode. Calls to set the driver to manual commit mode and to commit transactions are ignored. Calls to rollback a transaction cause the driver to return an error indicating that no transaction is started. Metadata indicates that the driver supports transactions and the ReadUncommitted transaction isolation level.

If set to 2 - Transactions, the data source and driver support manual transactions for supported data stores. Support for isolation levels depends on which backend data store is being used. If the data store does not support transactions (for example, Salesforce), then Transaction Mode is switched to 0 - No Transactions.
Trust Store

Attribute
TrustStore (TS)

Purpose
The location of the trust store file that contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication. The value can be a simple file name, or a relative path or absolute path. Relative paths are relative to the current directory. An absolute path is recommended, particularly if the current directory could change during the life of the application.

Valid values
path_name\trust_store_file_name
where:
path_name

  is the directory where the trust store file is located

trust_store_file_name

  is the name of the trust store file.

Default
None

GUI Tab
Security tab

See also
Data encryption

Trust Store Password

Attribute
TrustStorePassword (TSP)
**Purpose**
The password that is used to access the trust store file when server authentication is used. The trust store file contains a list of the Certificate Authorities (CAs) that the client trusts.

**Valid Values**

```
truststore_password
```

where:

```
truststore_password
```

is the password for the trust store file.

**Default**
None

**GUI Tab**
Security tab

**See also**
Data encryption

---

**User Name**

**Attribute**

LogonID (UID)

**Purpose**
Specifies the user ID for the Hybrid Data Pipeline connectivity service account. The user name is case-insensitive.

**Valid Values**

```
userid
```

where:

```
userid
```

is a valid user ID with permissions to access the Hybrid Data Pipeline connectivity service.

**Default**
None

**See also**

Logon Domain on page 687

**GUI Tab**
Security tab
Validate Server Certificate

Attribute

ValidateServerCertificate (VSC)

Purpose

Determines whether the connectivity service validates the certificate that is sent by the Hybrid Data Pipeline server when SSL encryption is enabled. When using SSL server authentication, any certificate sent by the Hybrid Data Pipeline server must be issued by a trusted Certificate Authority (CA). Disabling certificate validation reduces security by allowing man-in-the-middle (MITM) and other attacks. However, allowing the connectivity service to trust any certificate returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify trust store information on each client in the test environment.

Trust store information is specified using the Trust Store and Trust Store Password options.

Valid values

true | false

Behavior

If set to 1 (Enabled) or true, the connectivity service validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the trust store file. If the Host Name In Certificate option is specified, the connectivity service also validates the certificate using a host name. The Host Name In Certificate option provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the connectivity service is connecting to is the server that was requested.

If set to 0 (Disabled) or false, the connectivity service does not validate the certificate that is sent by the database server. The connectivity service ignores any trust store information specified by the Trust Store and Trust Store Password options.

Default

1 (Enabled)

GUI Tab

Security tab

See also

Data encryption

Varchar Threshold

Attribute

VarcharThreshold (VT)
Purpose

Specifies the threshold at which the driver describes columns of the data type SQL_VARCHAR as SQL_LONGVARCHAR. If the size of the SQL_VARCHAR column exceeds the value specified, the driver will describe the column as SQL_LONGVARCHAR when calling SQLDescribeCol and SQLColumns. This option allows you to fetch columns that would otherwise exceed the upper limit of the SQL_VARCHAR type for some third-party applications.

Valid values

x

where:

x

is the maximum size in characters of columns the driver will describe as SQL_VARCHAR.

Notes

• Configuring the Varchar Threshold and Min Long Varchar Size options allows you to fetch SQL_VARCHAR and SQL_LONGVARCHAR columns with sizes that fall between the data-type ranges used by some applications.

Default

None. If no value is specified, the driver does not change the described type for SQL_VARCHAR columns.

GUI tab

Advanced tab

See also

MinLongVarcharSize

WSRetryCount

Attribute

WSRetryCount (WSRC)

Purpose

The number of times the driver retries a timed-out Select request. Insert, Update, and Delete requests are never retried. The timeout period is specified by the WSTimeout (WST) connection option.

Valid Values

0 | x

where:

x

is a positive integer.
Behavior
If set to 0, the driver does not retry timed-out requests after the initial unsuccessful attempt.
If set to \( x \), the driver retries the timed-out request the specified number of times.

Default
3

GUI Tab
Web Service tab

See also
WSTimeout on page 699

WSTimeout

Attribute
WSTimeout (WST)

Purpose
Specifies the time, in seconds, that the driver waits for a response to a Web service request.

Valid Values
0 | \( x \)
where:
\( x \)

is a positive integer that defines the number of seconds the driver waits for a response to a Web service request.

Behavior
If set to 0, the driver waits indefinitely for a response; there is no timeout.
If set to \( x \), the driver uses the value as the default timeout for any statement created by the connection.
If a Select request times out and WSRetryCount (WSRC) is set to retry timed-out requests, the driver retries the request the specified number of times.

Default
120 (seconds)

GUI Tab
Web Service tab

See also
WSRetryCount on page 698
Application considerations

This section provides reference information for users of the Hybrid Data Pipeline for ODBC driver.

Verifying the driver version number

This section describes how to get version string information for the Hybrid Data Pipeline driver for ODBC and the Driver Manager.

Driver version string

The Hybrid Data Pipeline driver has a version string of the format:

\[XX.YY.ZZZZ\ (BAAAA, UBBBB)\]

where:

\[XX\]

is the major version of the driver.

\[YY\]

specifies the minor version of the driver.

\[ZZZZ\]

is the build number of the driver.

\[AAAA\]

is the build number of the driver's bas component.

\[BBBB\]

is the build number of the driver's utl component.

For example:

\[04.12.0034\ (B0005, U0006)\]

On Windows, you can check the version string through the properties of the driver DLL. Right-click the driver DLL and select Properties. The Properties dialog box appears. On the Version tab, click File Version in the Other version information list box.

You can always check the version string of a driver by looking at the About tab of the driver's Setup dialog.

On UNIX and Linux, you can check the version string by using the test loading tool shipped with the product. This tool, ivtestlib, is located in install_directory/bin.
The syntax for the tool is:

```plaintext
ivtestlib shared_object
```
or

```plaintext
ddtestlib shared_object
```

For example, for the 32-bit driver on Oracle Solaris:

```plaintext
ivtestlib ivhybrid01.so
```

returns:

04.12.0034 (B0005, U0006)

**Driver Manager version string (UNIX/Linux)**

**Note:** The driver uses the same Driver Manager as the Progress DataDirect Connect for ODBC drivers. For this reason, the Driver Manager version does not correspond to the version of the Hybrid Data Pipeline for ODBC driver.

The Driver Manager on UNIX and Linux has a version string of the format:

```
XX.YY.ZZZZ (UBBBB)
```

The component for the Unicode conversion tables (ICU) has a version string of the format:

```
XX.YY.ZZZZ
```

where:

- **XX** is the major version of the product.
- **YY** is the minor version of the product.
- **ZZZZ** is the build number of the driver or ICU component.
- **BBBB** is the build number of the product's utl component.

For example:

```
07.10.0001 (U0001)
```

<table>
<thead>
<tr>
<th>Driver</th>
<th>Utl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
On UNIX and Linux, you can check the version string by using the test loading tool shipped with the product. This tool, ivtestlib, is located in install_directory/bin.

The syntax for the tool is:

ivtestlib shared_object

or

ddtestlib shared_object

For example, for the 32-bit Driver Manager on Solaris:

ivtestlib libodbc.so

returns:

07.10.0001 (U0001)

For example, for the 64-bit Driver Manager on Solaris:

ddtestlib libodbc.so

returns:

07.10.0001 (U0001)

For example, for the 32-bit ICU component on Solaris:

ivtestlib libivicu27.so

returns:

07.10.0001

**Note:** On AIX, Linux, and Solaris, the full path to the product does not have to be specified for the test loading tool. The HP-UX version of the tool, however, requires the full path.

**getFileVersionString Function**

Version string information can also be obtained programmatically through the function `getFileVersionString`. This function can be used when the application is not directly calling ODBC functions.

This function is defined as follows and is located in each data source’s shared object:

```c
const unsigned char* getFileVersionString();
```

This function is prototyped in the `qesqlext.h` file shipped with the product.

**Retrieving data type information**

At times, you might need to get information about the data types that are supported by the cloud data store, for example, precision and scale. You can use the ODBC function `SQLGetTypeInfo` to do this.
On Windows, you can use ODBC Test to call SQLGetTypeInfo against the ODBC data source to return the data type information. See Troubleshooting on page 708 for details about ODBC Test.

On UNIX, Linux, or Windows, an application can call SQLGetTypeInfo. Here is an example of a C function that calls SQLGetTypeInfo and retrieves the information in the form of a SQL result set.

```c
void ODBC_GetTypeInfo(SQLHANDLE hstmt, SQLSMALLINT dataType)
{
    RETCODE rc;
    // There are 19 columns returned by SQLGetTypeInfo.
    // This example displays the first 3.
    // Check the ODBC 3.x specification for more information.

    // Variables to hold the data from each column
    char typeName[30];
    short sqlDataType;
    SQLINTEGER columnSize;
    SQLLEN strlenTypeName, strlenSqlDataType, strlenColumnSize;
    rc = SQLGetTypeInfo(hstmt, dataType);
    if (rc == SQL_SUCCESS) {
        // Bind the columns returned by the SQLGetTypeInfo result set.
        rc = SQLBindCol(hstmt, 1, SQL_C_CHAR, &typeName,
                        sizeof(typeName), &strlenTypeName);
        rc = SQLBindCol(hstmt, 2, SQL_C_SHORT, &sqlDataType,
                        sizeof(sqlDataType), &strlenSqlDataType);
        rc = SQLBindCol(hstmt, 3, SQL_C_LONG, &columnSize,
                        sizeof(columnSize), &strlenColumnSize);

        // Print column headings
        printf("TypeName  DataType  ColumnSize\n");
        printf("--------------------  ----------  ----------\n");
        do {
            // Fetch the results from executing SQLGetTypeInfo
            rc = SQLFetch(hstmt);
            if (rc == SQL_ERROR) {
                Procedure to retrieve errors from the SQLGetTypeInfo function
                ODBC_GetDiagRec(SQL_HANDLE_STMT, hstmt);
                break;
            }

            // Print the results
            printf("%-30s %10i %10u\n", typeName, sqlDataType, columnSize);
        } while (rc != SQL_NO_DATA);
    }
}
```

**Supported ODBC API functions**

The Hybrid Data Pipeline for odbc driver is Level 1 compliant; that is, it supports the Core and Level 1 ODBC conformance levels. It also supports a limited set of Level 2 functions, as described in the following tables.

**Table 135: Function Conformance for ODBC 2.x Applications**

<table>
<thead>
<tr>
<th>Core Functions</th>
<th>Level 1 Functions</th>
<th>Level 2 Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>• SQLAllocConnect</td>
<td>• SQLBrowseConnect</td>
<td>• SQLColumnPrivileges</td>
</tr>
<tr>
<td>Core Functions</td>
<td>Level 1 Functions</td>
<td>Level 2 Functions</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>• SQLAllocEnv</td>
<td>• SQLBulkOperations</td>
<td>• SQLDescribeParam</td>
</tr>
<tr>
<td>• SQLAllocStmt</td>
<td>• SQLDriverConnect</td>
<td>• SQLExtendedFetch (forward scrolling only)</td>
</tr>
<tr>
<td>• SQLBindCol</td>
<td>• SQLGetConnectOption</td>
<td>• SQLMoreResults</td>
</tr>
<tr>
<td>• SQLBindParameter</td>
<td>• SQLGetData</td>
<td>• SQLNativeSql</td>
</tr>
<tr>
<td>• SQLCancel</td>
<td>• SQLGetFunctions</td>
<td>• SQLNumParams</td>
</tr>
<tr>
<td>• SQLCloseCursor</td>
<td>• SQLGetTypeInfo</td>
<td>• SQLParamOptions</td>
</tr>
<tr>
<td>• SQLColAttribute</td>
<td>• SQLGetStmtOption</td>
<td>• SQLSetScrollOptions</td>
</tr>
<tr>
<td>• SQLColumns</td>
<td>• SQLParamData</td>
<td></td>
</tr>
<tr>
<td>• SQLConnect</td>
<td>• SQLPutData</td>
<td></td>
</tr>
<tr>
<td>• SQLCopyDesc</td>
<td>• SQSetConnectOption</td>
<td></td>
</tr>
<tr>
<td>• SQLDataSources</td>
<td>• SQLSetStmtOption</td>
<td></td>
</tr>
<tr>
<td>• SQLDescribeCol</td>
<td>• SQLSpecialColumns</td>
<td></td>
</tr>
<tr>
<td>• SQLDisconnect</td>
<td>• SQLStatistics</td>
<td></td>
</tr>
<tr>
<td>• SQLDrivers</td>
<td>• SQLTables</td>
<td></td>
</tr>
<tr>
<td>• SQLError</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLExecDirect</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLExecute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQFetch</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLFreeConnect</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLFreeEnv</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLFreeStmt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLGetCursorName</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLNumResultCols</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLPrepare</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLRowCount</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLSetCursorName</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SQLTransact</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The functions that the driver supports for ODBC 3.x are listed in the following table. Any additions to these supported functions or differences in the support of specific functions are listed in ODBC conformance level on page 707.
### Table 136: Function Conformance for ODBC 3.x Applications

<table>
<thead>
<tr>
<th>Function</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLAllocHandle</td>
<td>SQLGetInfo</td>
</tr>
<tr>
<td>SQLBindCol</td>
<td>SQLGetStmtAttr</td>
</tr>
<tr>
<td>SQLBindParameter</td>
<td>SQLGetTypeInfo</td>
</tr>
<tr>
<td>SQLBrowseConnect</td>
<td>SQLMoreResults</td>
</tr>
<tr>
<td>SQLBulkOperations</td>
<td>SQLNativeSql</td>
</tr>
<tr>
<td>SQLCancel</td>
<td>SQLNumParens</td>
</tr>
<tr>
<td>SQLCloseCursor</td>
<td>SQLNumResultCols</td>
</tr>
<tr>
<td>SQLColAttribute</td>
<td>SQLPrepare</td>
</tr>
<tr>
<td>SQLColumns</td>
<td>SQLPutData</td>
</tr>
<tr>
<td>SQLConnect</td>
<td>SQLRowCount</td>
</tr>
<tr>
<td>SQLCopyDesc</td>
<td>SQLRowCount</td>
</tr>
<tr>
<td>SQLDataSources</td>
<td>SQLSetConnectAttr</td>
</tr>
<tr>
<td>SQLDescribeCol</td>
<td>SQLSetCursorName</td>
</tr>
<tr>
<td>SQLDisconnect</td>
<td>SQLSetDescField</td>
</tr>
<tr>
<td>SQLDriverConnect</td>
<td>SQLSetDescRec</td>
</tr>
<tr>
<td>SQLDrivers</td>
<td>SQLSetEnvAttr</td>
</tr>
<tr>
<td>SQLEndTran</td>
<td>SQLSetStmtAttr</td>
</tr>
<tr>
<td>SQLError</td>
<td>SQLSpecialColumns</td>
</tr>
<tr>
<td>SQLExecDirect</td>
<td>SQLStatistics</td>
</tr>
<tr>
<td>SQLExecute</td>
<td>SQLTables</td>
</tr>
<tr>
<td>SQLExtendedFetch</td>
<td>SQLTransact</td>
</tr>
<tr>
<td>SQLFetch</td>
<td></td>
</tr>
<tr>
<td>SQLFetchScroll (forward scrolling only)</td>
<td></td>
</tr>
<tr>
<td>SQLFreeHandle</td>
<td></td>
</tr>
<tr>
<td>SQLFreeStmt</td>
<td></td>
</tr>
<tr>
<td>SQLGetConnectAttr</td>
<td></td>
</tr>
<tr>
<td>SQLGetCursorName</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SQLGetData</td>
</tr>
<tr>
<td></td>
<td>SQLGetDescField</td>
</tr>
<tr>
<td></td>
<td>SQLGetDescRec</td>
</tr>
<tr>
<td></td>
<td>SQLGetDiagField</td>
</tr>
<tr>
<td></td>
<td>SQLGetDiagRec</td>
</tr>
<tr>
<td></td>
<td>SQLGetEnvAttr</td>
</tr>
<tr>
<td></td>
<td>SQLGetFunctions</td>
</tr>
<tr>
<td></td>
<td>SQLGetInfo</td>
</tr>
<tr>
<td></td>
<td>SQLGetStmtAttr</td>
</tr>
<tr>
<td></td>
<td>SQLGetTypeInfo</td>
</tr>
<tr>
<td></td>
<td>SQLMoreResults</td>
</tr>
<tr>
<td></td>
<td>SQLNativeSql</td>
</tr>
<tr>
<td></td>
<td>SQLNumParens</td>
</tr>
<tr>
<td></td>
<td>SQLNumResultCols</td>
</tr>
<tr>
<td></td>
<td>SQLPrepare</td>
</tr>
<tr>
<td></td>
<td>SQLPutData</td>
</tr>
<tr>
<td></td>
<td>SQLRemoveFunction</td>
</tr>
<tr>
<td></td>
<td>SQLTransact</td>
</tr>
</tbody>
</table>
SQLCancel

The Hybrid Data Pipeline Driver for ODBC supports SQLCancel, which can be used to stop the execution of a statement, including any processing being done by the Hybrid Data Pipeline connectivity service. Unlike when using SQLFreeStmt (SQL_CLOSE), any results generated before SQLCancel is called are still available for retrieval. Fetching past the generated results will return a statement-was-cancelled error. Refer to the ODBC specification for details on the usage of SQLCancel.

Note: Because the Hybrid Data Pipeline connectivity service may be accumulating results for a statement, canceling a statement is usually treated as if a function is running on another thread. Only if all results from the statement have been retrieved will it be considered that no processing is being done on the statement.

Scalar functions

This section lists the scalar functions that ODBC supports. Any given data store may not support all these functions. To check which scalar functions are supported by a driver, use the SQLGetInfo ODBC function. Refer to the documentation for your data store to find out which functions are supported, and to the Microsoft ODBC Programmer’s Reference descriptions of the functions.

You can use these scalar functions in SQL statements using the following syntax:

```
(fn scalar-function)
```

where `scalar-function` is one of the functions listed in the following tables. For example:

```
SELECT {fn UCASE(NAME)} FROM EMP
```

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>CURRENT_USER</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAYNAME</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ATAN2</td>
<td>DAYOFMONTH</td>
<td>USER</td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITAND</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITOR</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>CEILING</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>String Functions</td>
<td>Numeric Functions</td>
<td>Timedate Functions</td>
<td>System Functions</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------</td>
<td>-------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>FLOOR</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>LOG10</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>RAWTOHEX</td>
<td>MOD</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>PI</td>
<td>CURRENT_DATE</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>POWER</td>
<td>CURRENT_TIME</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>RADIANS</td>
<td>CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOUNDINDEX</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>ROUNDMAGIC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTR</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>SORT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPPER</td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TRUNCATE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For more information about ODBC data types, refer to the Microsoft ODBC Programmer's Reference.

**ODBC conformance level**

The Hybrid Data Pipeline driver supports the following Level 2 functions:

- SQLColumnPrivileges
- SQLDescribeParam
- SQLForeignKeys
- SQLTablePrivileges

See Supported ODBC API functions on page 703 for a list of the Core and Level 1 functions supported by the driver.
Troubleshooting

This section provides tips for troubleshooting and discusses log files and diagnostic tools.

Determining where an issue originates

If issues arise while using Hybrid Data Pipeline Driver for ODBC, it will be helpful to first narrow down the origin of the problem. This section describes three types of issues, provides some typical causes of the issues, lists some diagnostic tools that are useful to troubleshoot the issues, and, in some cases, provides suggestions for resolving the issues.

Setup/connection issues

Setup and connection issues can cause hangs during connection or configuration. Some common errors that are returned by the ODBC driver if you are experiencing a setup or connection issue include:

- Specified driver could not be loaded.
- Data source name not found and no default driver specified.
- Cannot open shared library: libodbc.sl.
- Invalid user ID or password: Unable to connect to destination.
- INVALID_LOGIN: Invalid username, password, security token; or user locked out.

See Required properties on page 661 for a list of connection string attributes that are required for the driver.

For UNIX and Linux users: See Test loading tools for UNIX and Linux on page 711 for information about a helpful diagnostic tool.

Interoperability issues

Interoperability issues can occur with a working ODBC application in any of the following ODBC components: ODBC application, ODBC driver, ODBC Driver Manager, and/or data source.

For example, any of the following problems may occur because of an interoperability issue:

- SQL statements may fail to execute.
- Data may be returned/updated/deleted/inserted incorrectly.
- A hang or core dump may occur.

Isolate the component in which the issue is occurring. Is it an ODBC application, an ODBC driver, an ODBC Driver Manager, or a data source issue?

To troubleshoot the issue:

1. Test to see if your ODBC application is the source of the problem. To do this, replace your working ODBC application with a more simple application. If you can reproduce the issue, you know your ODBC application is not the cause.
On Windows, you can use ODBC Test, which is part of the Microsoft ODBC SDK, or the example application that is shipped with the driver. See Creating a trace log on page 714 for details.

2. If neither the ODBC application nor the data source is the source of your problem, troubleshoot the ODBC driver and the ODBC Driver Manager.

   In this case, we recommend that you create an ODBC trace log to provide to Technical Support. See ODBC Test on page 715 for details.

Performance issues

Developing performance-oriented ODBC applications requires iteration and perseverance. You must be willing to change your application and test it to see if your changes helped performance. Microsoft's ODBC Programmer’s Reference does not provide information about system performance. In addition, ODBC drivers and the ODBC Driver Manager do not return warnings when applications run inefficiently.

Some general guidelines for developing performance-oriented ODBC applications include:

- Use catalog functions appropriately.
- Retrieve only required data.
- Select functions that optimize performance.
- Manage connections and updates.

Log files created during installation or upgrade

If you encounter problems during an installation or upgrade, check the log files. During an installation or upgrade, the installer creates the following log files:

- If the installer successfully creates the product installation directory, the installer generates the following log files.
  - `<install_dir>/ProgressHIDPServer_installation/Logs/Progress_DataDirect_Hybrid_Data_Pipeline_Server_Install_<date>.log`
  - `<install_dir>/ddcloud/deploy.log`
  - `<install_dir>/ddcloud/final.log`

- If the installation or upgrade fails such that the installer does not create the installation directory, the installer writes a file named `Progress__DataDirect_Hybrid_Data_Pipeline_Server_InstallFailed.txt` to the machine’s default temporary directory.

- When a silent installation or upgrade fails but an installation directory was created, the installer writes a file named `SilentInstallerError.log` to the home directory of the target machine.

- When validation for a silent installation fails, the installer generates the file `SilentInstallInfo.log` in the home directory of the target machine but completes a full installation.

If you need help interpreting these files, contact Progress DataDirect Technical Support.

Error message syntax

Any of the following components can generate errors:
• Hybrid Data Pipeline Driver for ODBC
• Hybrid Data Pipeline connectivity service
• A Hybrid Data Pipeline data store
• ODBC Driver Manager

When troubleshooting, it is helpful to know where the error originated. The following topics provide the syntax and an example for each error type.

Hybrid Data Pipeline driver errors

Syntax

[vendor][ODBC_component] message

Example

[DataDirect][ODBC Hybrid driver] Object has been closed.

See also
Check the last ODBC call made by your application for possible problems or contact your ODBC application vendor.

Service errors

Service errors come from the Hybrid Data Pipeline connectivity service.

[vendor][ODBC_component][Service] message

Example

[DataDirect][ODBC Hybrid Driver][Service] Invalid user ID or password.

Driver Manager errors (Windows)

On Windows, the Microsoft Driver Manager is a DLL that establishes connections with drivers, submits requests to drivers, and returns results to applications.

Syntax

[vendor][ODBCXXX] message

Example

[Microsoft][ODBC Driver Manager] Driver does not support this function
If you receive this type of error, consult the *Programmer’s Reference* for the Microsoft ODBC Software Development Kit available from Microsoft.

**Data Store errors**

**Syntax**

```
[vendor][ODBC_component] [data_store] message
```

**Example**

```
[DataDirect][ODBC Hybrid Driver][SalesForce] Table not found in statement
```

**See also**

You may need to check your data store documentation for more information or consult your data administrator.

**Driver Manager errors (UNIX and Linux)**

On UNIX and Linux, the Driver Manager is provided by Progress DataDirect.

**Syntax**

```
[vendor][ODBCXXX] message
```

**Example**

```
[DataDirect][ODBC lib] String data code page conversion failed.
```

UNIX and Linux error handling follows the X/Open XPG3 messaging catalog system. Localized error messages are stored in the subdirectory:

```
locale/localized_territory_directory/LC_MESSAGES
```

where:

```
localized_territory_directory
```

depends on your language. For instance, German localization files are stored in `locale/de/LC_MESSAGES`, where `de` is the locale for German.

**Test loading tools for UNIX and Linux**

The `ivtestlib` (32-bit drivers) and `ddtestlib` (64-bit drivers) test loading tools are provided to test load drivers and help diagnose configuration problems in the UNIX and Linux environments. Such problems might include environment variables not correctly set. This tool is installed in the `/bin` subdirectory in the product installation directory. It attempts to load a specified ODBC driver and prints out all available error information if the load fails.
For example, if the drivers are installed in `/opt/odbc/lib`, the following command attempts to load the 32-bit Hybrid Data Pipeline driver on Solaris, where `nn` represents the version number of the driver:

```
ivtestlib /opt/odbc/lib/ivhybridnn.so
```

**Note:** On Solaris, AIX, and Linux, the full path to the driver does not have to be specified for the tool. The HP-UX version, however, requires the full path.

If the load is successful, the tool returns a success message along with the version string of the driver. If the driver cannot be loaded, the tool returns an error message explaining why.

See [Verifying the driver version number](#) on page 700 for details about version strings.

The next step is to configure a data source through the system information file.

### ODBC Trace

ODBC tracing allows you to trace calls to ODBC drivers and create a log of the traces. Progress DataDirect provides a tracing library that is enhanced to operate more efficiently, especially in production environments, where log files can rapidly grow in size. The DataDirect tracing library allows you to control the size and number of log files.

See [Error message syntax](#) on page 709 for a description of the different types of errors that can be logged.

### Enabling tracing on Windows Systems

On Windows, open the ODBC Data Source Administrator and select the Tracing tab. To specify the path and name of the trace log file, type the path and name in the Log File Path field or click **Browse** to select a log file. If no location is specified, the trace log resides in the working directory of the application you are using.

Click **Select DLL** in the Custom Trace DLL pane to select the DataDirect enhanced tracing library, `xxtrcnn.dll`, where `xx` represents either `iv` (32-bit version) or `dd` (64-bit version), and `nn` represents the driver level number, for example, `ivtrc27.dll`. The library is installed in the `\Windows\System32` directory.

After making changes on the Tracing tab, click **Apply** for them to take effect.

Enable tracing by clicking **Start Tracing Now**. Tracing continues until you disable it by clicking **Stop Tracing Now**. Be sure to turn off tracing when you are finished reproducing the issue because tracing decreases the performance of your ODBC application.

When tracing is enabled, information is written to the following trace log files:

- **Trace log file** *(trace_filename.log)* in the specified directory.
- **Trace information log file** *(trace_filenameINFO.log)*. This file is created in the same directory as the trace log file and logs the following SQLGetInfo information:
  - `SQL_DBMS_NAME`
  - `SQL_DBMS_VER`
  - `SQL_DRIVER_NAME`
  - `SQL_DRIVER_VER`
  - `SQL_DEFAULT_TXN_ISOLATION`
Configuring trace files on Windows Systems

The DataDirect enhanced tracing library allows you to control the size and number of log files. The file size limit of the log file (in KB) is specified by the Windows Registry key ODBCTraceMaxFileSize. Once the size limit is reached, a new log file is created and logging continues in the new file until it reaches its file size limit, after which another log file is created, and so on.

The maximum number of files that can be created is specified by the Registry key ODBCTraceMaxNumFiles. Once the maximum number of log files is created, tracing reopens the first file in the sequence, deletes the content, and continues logging in that file until the file size limit is reached, after which it repeats the process with the next file in the sequence. Subsequent files are named by appending sequential numbers, starting at 1 and incrementing by 1, to the end of the original file name, for example, SQL1.LOG, SQL2.LOG, and so on.

The default values of ODBCTraceMaxFileSize and ODBCTraceMaxNumFiles are 102400 KB and 10, respectively. To change these values, add or modify the keys in the following Windows Registry section:

```
[HKEY_CURRENT_USER\SOFTWARE\ODBC\ODBC.INI\ODBC]
```

⚠️ Warning: Do not edit the Registry unless you are an experienced user. Consult your system administrator if you have not edited the Registry before.

Edit each key using your values and close the Registry.

Enabling and configuring tracing on UNIX/Linux systems

The [ODBC] section of the system information file includes several keywords that control tracing:

- **Trace=[0 | 1]**
- **TraceFile=trace_filename**
- **TraceDll=ODBCHOME/lib/xxtrcnn.zz**
- **ODBCTraceMaxFileSize=file_size**
- **ODBCTraceMaxNumFiles=file_number**
- **TraceOptions=0**

where:

**Trace=[0 | 1]**

Allows you to enable tracing by setting the value of Trace to 1. Disable tracing by setting the value to 0 (the default). Tracing continues until you disable it. Be sure to turn off tracing when you are finished reproducing the issue because tracing decreases the performance of your ODBC application.

**TraceFile=trace_filename**

Specifies the path and name of the trace log file. If no path is specified, the trace log resides in the working directory of the application you are using.

**TraceDll=ODBCHOME/lib/xxtrcnn.zz**

Specifies the library to use for tracing. The driver installation includes a DataDirect enhanced library to perform tracing, xxtrcnn.zz, where xx represents either iv (32-bit version) or dd (64-bit version), nn represents the driver level number, and zz represents either so or sl. For example, ivtrc27.so is the 32-bit version of the library. To use a custom shared library instead, enter the path and name of the library as the value for the TraceDll keyword.
The DataDirect enhanced tracing library allows you to control the size and number of log files with the ODBCTraceMaxFileSize and ODBCTraceMaxNumFiles keywords.

**ODBCTraceMaxFileSize**=

The ODBCTraceMaxFileSize keyword specifies the file size limit (in KB) of the log file. Once this file size limit is reached, a new log file is created and logging continues in the new file until it reaches the file size limit, after which another log file is created, and so on. The default is 102400.

**ODBCTraceMaxNumFiles**=

The ODBCTraceMaxNumFiles keyword specifies the maximum number of log files that can be created. The default is 10. Once the maximum number of log files is created, tracing reopens the first file in the sequence, deletes the content, and continues logging in that file until the file size limit is reached, after which it repeats the process with the next file in the sequence. Subsequent files are named by appending sequential numbers, starting at 1 and incrementing by 1, to the end of the original file name, for example, odbctrace1.out, odbctrace2.out, and so on.

**TraceOptions**=[0 | 1 | 2 | 3]

The ODBCTraceMaxNumFiles keyword specifies whether to print the current timestamp, parent process id, process id, and thread id for all ODBC functions to the output file. The default is 0.

- If set to 0, the driver uses standard ODBC tracing.
- If set to 1, the log file includes a timestamp on ENTRY and EXIT of each ODBC function.
- If set to 2, the log file prints a header on every line. By default, the header includes the parent process ID and process ID.
- If set to 3, both TraceOptions=1 and TraceOptions=2 are enabled. The header includes a timestamp as well as a parent process ID and process ID.

In the following example of trace settings, tracing has been enabled, the name of the log file is odbctrace.out, the maximum size of the log file is 51200 KB, and the maximum number of log files is 8. The library for tracing is ivtrcnn.so, where nn is the driver level number. Timestamp and other information is included in odbctrace.out.

```
Trace=1
TraceFile=ODBCHOME/lib/odbctrace.out
TraceDll=ODBCHOME/lib/ivtrcnn.so
ODBCTraceMaxFileSize=51200
ODBCTraceMaxNumFiles=8
TraceOptions=3
```

### Creating a trace log

Creating a trace log is particularly useful when you are troubleshooting an issue.

**To create a trace log:**

1. Enable tracing:
   - On Windows, enable tracing through the Tracing tab of the ODBC Data Source Administrator.
   - On UNIX and Linux, enable tracing by directly modifying the [ODBC] section in the system information (odbc.ini) file.

2. Start the ODBC application and reproduce the issue.
3. Stop the application and turn off tracing.
4. Open the log file in a text editor and review the output to help you debug the problem.

For a complete explanation of tracing, refer to the following Progress DataDirect Knowledgebase document: http://knowledgebase.progress.com/articles/Article/3049

Other tools

The Progress DataDirect Support Web site provides other diagnostic tools that you can download to assist you with troubleshooting. These tools are not shipped with the product. Refer to the Progress DataDirect Web page: https://www.progress.com/support/evaluation/download-resources/download-tools

Progress DataDirect also provides a Knowledgebase that is useful in troubleshooting problems.

ODBC Test

On Windows, Microsoft® ships with its ODBC SDK, an ODBC-enabled application, named ODBC Test, that you can use to test ODBC drivers and the ODBC Driver Manager. ODBC 3.51 includes both ANSI and Unicode-enabled versions of ODBC Test.

To use ODBC Test, you must understand the ODBC API, the C language, and SQL. For more information about ODBC Test, refer to the Microsoft ODBC SDK Guide.

Using the Driver with Microsoft Access

Progress DataDirect has included non-standard connection options (workarounds) for the Hybrid Data Pipeline for ODBC driver that enable you to take full advantage of packaged ODBC-enabled applications requiring non-standard or extended behavior.

When using the Hybrid Data Pipeline Driver for ODBC with Microsoft Access, we recommend that you create a separate user data source that includes the following two workarounds.

WorkAvoids=16777216
WorkAvoids2=8192

See WorkAround options on page 726 for more information on using workarounds.

Internationalization, localization, and Unicode

Hybrid Data Pipeline Driver for ODBC is a Unicode driver. This section provides an overview of how internationalization, localization, and Unicode relate to each other, describe the background of Unicode, and explain how Unicode drivers process Unicode data and encodings.

Internationalization and Localization

Software that has been designed for internationalization is able to manage different linguistic and cultural conventions transparently and without modification. The same binary copy of an application should run on any localized version of an operating system without requiring source code changes.
Software that has been designed for localization includes language translation (such as text messages, icons, and buttons), cultural data (such as dates, times, and currency), and other components (such as input methods and spell checkers) for meeting regional market requirements.

Properly designed applications can accommodate a localized interface without extensive modification. The applications can be designed, first, to run internationally, and, second, to accommodate the language- and cultural-specific elements of a designated locale.

**Locale**

A locale represents the language and cultural data chosen by the user and dynamically loaded into memory at runtime. The locale settings are applied to the operating system and to subsequent application launches.

While language is a fairly straightforward item, cultural data is a little more complex. Dates, numbers, and currency are all examples of data that is formatted according to cultural expectations. Because cultural preferences are bound to a geographic area, country is an important element of locale. Together these two elements (language and country) provide a precise context in which information can be presented. Locale presents information in the language and form that is best understood and appreciated by the local user.

**Language**

A locale's language is specified by the ISO 639 standard. The following table lists some commonly used language codes.

**Table 138: Language Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>en</td>
<td>English</td>
</tr>
<tr>
<td>nl</td>
<td>Dutch</td>
</tr>
<tr>
<td>fr</td>
<td>French</td>
</tr>
<tr>
<td>es</td>
<td>Spanish</td>
</tr>
<tr>
<td>en</td>
<td>English</td>
</tr>
<tr>
<td>zh</td>
<td>Chinese</td>
</tr>
<tr>
<td>ja</td>
<td>Japanese</td>
</tr>
<tr>
<td>vi</td>
<td>Vietnamese</td>
</tr>
</tbody>
</table>

Because language is correlated with geography, a language code might not capture all the nuances of usage in a particular area. For example, French and Canadian French may use different phrases and terms to mean different things even though basic grammar and vocabulary are the same. Language is only one element of locale.
**Variant**

A variant is an optional extension to a locale. It identifies a custom locale that is not possible to create with just language and country codes. Variants can be used by anyone to add additional context for identifying a locale. For example, the locale en_US represents English (United States), but en_US_CA represents even more information and might identify a locale for English (California, U.S.A). Operating system or software vendors can use these variants to create more descriptive locales for their specific environments.

**Country**

The locale's country identifier is also specified by an ISO standard, ISO 3166, which describes valid two-letter codes for all countries. ISO 3166 defines these codes in uppercase letters. The following table lists some commonly used country codes.

<table>
<thead>
<tr>
<th>Country Code</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>US</td>
<td>United States</td>
</tr>
<tr>
<td>FR</td>
<td>France</td>
</tr>
<tr>
<td>IE</td>
<td>Ireland</td>
</tr>
<tr>
<td>CA</td>
<td>Canada</td>
</tr>
<tr>
<td>MX</td>
<td>Mexico</td>
</tr>
</tbody>
</table>

The country code provides more contextual information for a locale and affects a language's usage, word spelling, and collation rules.

**Unicode Character Encoding**

In addition to locale, the other major component of internationalizing software is the use of the Universal Codeset, or Unicode. Most developers know that Unicode is a standard encoding that can be used to support multilingual character sets. Unfortunately, understanding Unicode is not as simple as its name would indicate. Software developers have used a number of character encodings, from ASCII to Unicode, to solve the many problems that arise when developing software applications that can be used worldwide.

**Background**

Most legacy computing environments have used ASCII character encoding developed by the ANSI standards body to store and manipulate character strings inside software applications. ASCII encoding was convenient for programmers because each ASCII character could be stored as a byte. The initial version of ASCII used only 7 of the 8 bits available in a byte, which meant that applications could use only 128 different characters. This version of ASCII could not account for European characters and was completely inadequate for Asian characters. Using the eighth bit to extend the total range of characters to 256 added support for most European characters. Today, ASCII refers to either the 7-bit or 8-bit encoding of characters.
As the need increased for applications with additional international support, ANSI again increased the functionality of ASCII by developing an extension to accommodate multilingual software. The extension, known as the Double-Byte Character Set (DBCS), allowed existing applications to function without change, but provided for the use of additional characters, including complex Asian characters. With DBCS, characters map to either one byte (for example, American ASCII characters) or two bytes (for example, Asian characters). The DBCS environment also introduced the concept of an operating system code page that identified how characters would be encoded into byte sequences in a particular computing environment. DBCS encoding provided a cross-platform mechanism for building multilingual applications.

Using a DBCS, however, was not ideal; many developers felt that there was a better way to solve the problem. A group of leading software companies joined forces to form the Unicode Consortium. Together, they produced a new solution to building worldwide applications—Unicode. Unicode was originally designed as a fixed-width, uniform two-byte designation that could represent all modern scripts without the use of code pages. The Unicode Consortium has continued to evaluate new characters, and the current number of supported characters is over 112,000.

Although it seemed to be the perfect solution to building multilingual applications, Unicode started off with a significant drawback—it would have to be retrofitted into existing computing environments. To use the new paradigm, all applications would have to change. As a result, several standards-based transliterations were designed to convert two-byte fixed Unicode values into more appropriate character encodings, including, among others, UTF-8, UCS-2, and UTF-16.

UTF-8 is a standard method for transforming Unicode values into byte sequences that maintain transparency for all ASCII codes. UTF-8 is recognized by the Unicode Consortium as a mechanism for transforming Unicode values and is popular for use with HTML, XML, and other protocols. UTF-8 is, however, currently used primarily on AIX, HP-UX, Solaris, and Linux.

UCS-2 encoding is a fixed, two-byte encoding sequence and is a method for transforming Unicode values into byte sequences. It is the standard for Windows 95, Windows 98, Windows Me, and Windows NT.

UTF-16 is a superset of UCS-2, with the addition of some special characters in surrogate pairs. UTF-16 is the standard encoding for Windows 2000, Windows XP, Windows Server 2003 and higher, Windows Vista, Windows 7, and higher. Microsoft recommends using UTF-16 for new applications.

Hybrid Data Pipeline Driver for ODBC is fully Unicode enabled. On UNIX and Linux platforms, the driver supports both UTF-8 and UTF-16. On Windows platforms, the driver supports UCS-2/UTF-16 only.

**Unicode Support in Databases**

Many database vendors support Unicode data types natively in their systems. With Unicode support, one database can hold multiple languages. For example, a large multinational corporation could store expense data in the local languages for the Japanese, U.S., English, German, and French offices in one database.

Not surprisingly, the implementation of Unicode data types varies from vendor to vendor. For example, the Microsoft SQL Server 2012 implementation of Unicode provides data in UTF-16 format, while Oracle provides Unicode data types in UTF-8 and UTF-16 formats. A consistent implementation of Unicode not only depends on the operating system, but also on the database itself.

**Unicode Support in ODBC**

Prior to the ODBC 3.5 standard, all ODBC access to function calls and string data types was through ANSI encoding (either ASCII or DBCS). Applications and drivers were both ANSI-based.

The ODBC 3.5 standard specified that the ODBC Driver Manager (on both Windows and UNIX) be capable of mapping both Unicode function calls and string data types to ANSI encoding as transparently as possible. This meant that ODBC 3.5-compliant Unicode applications could use Unicode function calls and string data types with ANSI drivers because the Driver Manager could convert them to ANSI. Because of character limitations in ANSI, however, not all conversions are possible.
The ODBC Driver Manager version 3.5 and later, therefore, supports the following configurations:

- ANSI application with a Unicode driver
- ANSI application with an ANSI driver
- Unicode application with a Unicode driver
- Unicode application with an ANSI driver

A Unicode application can work with an ANSI driver because the Driver Manager provides limited Unicode-to-ANSI mapping. The Driver Manager makes it possible for a pre-3.5 ANSI driver to work with a Unicode application. What distinguishes a Unicode driver from a non-Unicode driver is the Unicode driver’s capacity to interpret Unicode function calls without the intervention of the Driver Manager, as described in the following section.

Unicode ODBC Drivers

The way in which a driver handles function calls from a Unicode application determines whether it is considered a "Unicode driver."

Instead of the standard ANSI SQL function calls, such as SQLConnect, Unicode applications use "W" (wide) function calls, such as SQLConnectW. Hybrid Data Pipeline Driver for ODBC supports "W" function calls, so the Driver Manager can pass them through to the driver without conversion to ANSI.

For Hybrid Data Pipeline Driver for ODBC on UNIX and Linux, the Driver Manager determines the type of Unicode encoding of both the application and the driver, and performs conversions when the application and driver use different types of encoding. This determination is made by checking two ODBC environment attributes: SQL_ATTR_APP_UNICODE_TYPE and SQL_ATTR_DRIVER_UNICODE_TYPE. DriverManager and Unicode Encoding on UNIX/Linux on page 721 describes in detail how this is done.

Unicode Application with a Unicode Driver

An operation involving a Unicode application and a Unicode driver that use the same Unicode encoding is efficient because no function conversion is involved. If the application and the driver each use different types of encoding, there is some conversion overhead. See DriverManager and Unicode Encoding on UNIX/Linux on page 721 for details.

Windows

1. The Unicode application sends UCS-2 or UTF-16 function calls to the Driver Manager.
2. The Driver Manager does not have to convert the UCS-2/UTF-16 function calls to ANSI. It passes the Unicode function call to the Unicode driver.
3. The driver returns UCS-2/UTF-16 argument values to the Driver Manager.
4. The Driver Manager returns UCS-2/UTF-16 function calls to the application.

UNIX and Linux

1. The Unicode application sends function calls to the Driver Manager. The Driver Manager expects these function calls to be UTF-8 or UTF-16 based on the value of the SQL_ATTR_APP_UNICODE_TYPE attribute.
2. The Driver Manager passes Unicode function calls to the Unicode driver. The Driver Manager has to perform function call conversions if the SQL_ATTR_APP_UNICODE_TYPE is different from the SQL_ATTR_DRIVER_UNICODE_TYPE.
3. The driver returns argument values to the DriverManager. Whether these are UTF-8 or UTF-16 argument values is based on the value of the SQL_ATTR_DRIVER_UNICODE_TYPE attribute.

4. The DriverManager returns appropriate function calls to the application based on the SQL_ATTR_APP_UNICODE_TYPE attribute value. The DriverManager has to perform function call conversions if the SQL_ATTR_DRIVER_UNICODE_TYPE value is different from the SQL_ATTR_APP_UNICODE_TYPE value.

Data

ODBC C data types are used to indicate the type of C buffers that store data in the application. This is in contrast to SQL data types, which are mapped to native database types to store data in a database (data store). ANSI applications bind to the C data type SQL_C_CHAR and expect to receive information bound in the same way. Similarly, most Unicode applications bind to the C data type SQL_C_WCHAR (wide data type) and expect to receive information bound in the same way. Any ODBC 3.5-compliant Unicode driver must be capable of supporting SQL_C_CHAR and SQL_C_WCHAR so that it can return data to both ANSI and Unicode applications.

When the driver communicates with the database, it must use ODBC SQL data types, such as SQL_CHAR and SQL_WCHAR, that map to native database types. In the case of ANSI data and an ANSI database, the driver receives data bound to SQL_C_CHAR and passes it to the database as SQL_CHAR. The same is true of SQL_C_WCHAR and SQL_WCHAR in the case of Unicode data and a Unicode database.

When data from the application and the data stored in the database differ in format, for example, ANSI application data and Unicode database data, conversions must be performed. The driver cannot receive SQL_C_CHAR data and pass it to a Unicode database that expects to receive a SQL_WCHAR data type. The driver or the DriverManager must be capable of converting SQL_C_CHAR to SQL_WCHAR, and vice versa.

The simplest cases of data communication are when the application, the driver, and the database are all of the same type and encoding, Unicode-to-Unicode-to-Unicode. There is no data conversion involved in these instances.

When a difference exists between data types, a conversion from one type to another must take place at the driver or DriverManager level, which involves additional overhead. The type of driver determines whether these conversions are performed by the driver or the DriverManager. DriverManager and Unicode Encoding on UNIX/Linux on page 721 describes how the DriverManager determines the type of Unicode encoding of the application and driver.

The Unicode driver, not the DriverManager, must convert SQL_C_CHAR (ANSI) data to SQL_WCHAR (Unicode) data, and vice versa, as well as SQL_C_WCHAR (Unicode) data to SQL_CHAR (ANSI) data, and vice versa.

The driver must use client code page information (Active Code Page on Windows and IANAAppCodePage attribute on UNIX/Linux) to determine which ANSI code page to use for the conversions. The Active Code Page or IANAAppCodePage must match the database default character encoding; if it does not, conversion errors are possible. How an individual driver exchanges different types of data with a particular database at the database level is beyond the scope of this discussion.

Default Unicode mapping

The default Unicode mapping for an application’s SQL_C_WCHAR variable is:

<table>
<thead>
<tr>
<th>Platform</th>
<th>Default Unicode Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>UCS-2/UTF-16</td>
</tr>
<tr>
<td>AIX</td>
<td>UTF-8</td>
</tr>
<tr>
<td>HP-UX</td>
<td>UTF-8</td>
</tr>
</tbody>
</table>
### Default Unicode Mapping

<table>
<thead>
<tr>
<th>Platform</th>
<th>Default Unicode Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solaris</td>
<td>UTF-8</td>
</tr>
<tr>
<td>Linux</td>
<td>UTF-8</td>
</tr>
</tbody>
</table>

### Connection Attribute for Unicode

If you do not want to use the default Unicode mappings for `SQL_C_WCHAR`, a connection attribute is available to override the default mappings. This attribute determines how character data is converted and presented to an application and the database.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL_ATTR_APP_WCHAR_TYPE (1061)</td>
<td>Sets the <code>SQL_C_WCHAR</code> type for parameter and column binding to the Unicode type, either <code>SQL_DD_CP_UTF16</code> (default for Windows) or <code>SQL_DD_CP_UTF8</code> (default for UNIX/Linux).</td>
</tr>
</tbody>
</table>

You can set this attribute before or after you connect. After this attribute is set, all conversions are made based on the character set specified.

For example:

```c
rc = SQLSetConnectAttr (hdbc, SQL_ATTR_APP_WCHAR_TYPE, (void *)SQL_DD_CP_UTF16, SQL_IS_INTEGER);
```

`SQLGetConnectAttr` and `SQLSetConnectAttr` for the `SQL_ATTR_APP_WCHAR_TYPE` attribute return a SQL State of HYC00 for drivers that do not support Unicode.

This connection attribute and its valid values can be found in the file `qesqlext.h`, which is installed with the product.

### Driver Manager and Unicode Encoding on UNIX/Linux

Unicode ODBC drivers on UNIX and Linux can use UTF-8 or UTF-16 encoding. This would normally mean that a UTF-8 application could not work with a UTF-16 driver, and, conversely, that a UTF-16 application could not work with a UTF-8 driver. To accomplish the goal of being able to use a single UTF-8 or UTF-16 application with either a UTF-8 or UTF-16 driver, the Driver Manager must be able to determine with which type of encoding the application and driver use and, if necessary, convert them accordingly.

To make this determination, the Driver Manager supports two ODBC environment attributes:

- `SQL_ATTR_APP_UNICODE_TYPE` and `SQL_ATTR_DRIVER_UNICODE_TYPE`, each with possible values of `SQL_DD_CP_UTF8` and `SQL_DD_CP_UTF16`. The default value is `SQL_DD_CP_UTF8`.

The Driver Manager performs the following steps before actually connecting to the driver.

1. **Determine the application Unicode type:** Applications that use UTF-16 encoding for their string types need to set `SQL_ATTR_APP_UNICODE_TYPE` accordingly before connecting to any driver. When the Driver Manager reads this attribute, it expects all string arguments to the ODBC "W" functions to be in the specified Unicode format. This attribute also indicates how the `SQL_C_WCHAR` buffers must be encoded.

2. **Determine the driver Unicode type:** The Driver Manager must determine through which Unicode encoding the driver supports its "W" functions. This is done as follows:
a. SQLGetEnvAttr(SQL_ATTR_DRIVER_UNICODE_TYPE) is called in the driver by the Driver Manager. The driver returns either SQL_DD_CP_UTF16 or SQL_DD_CP_UTF8 to indicate to the Driver Manager which encoding it expects.

b. If the preceding call to SQLGetEnvAttr fails, the Driver Manager looks either in the Data Source section of the odbc.ini specified by the connection string or in the connection string itself for a connection option named DriverUnicodeType. Valid values for this option are 1 (UTF-16) or 2 (UTF-8). The Driver Manager assumes that the Unicode encoding of the driver corresponds to the value specified.

c. If neither of the preceding attempts are successful, the Driver Manager assumes that the Unicode encoding of the driver is UTF-8.

3. Determine if the driver supports SQL_ATTR_WCHAR_TYPE: SQLSetConnectAttr
(SQL_ATTR_WCHAR_TYPE, x) is called in the driver by the Driver Manager, where x is either SQL_DD_CP_UTF8 or SQL_DD_CP_UTF16, depending on the value of the SQL_ATTR_APP_UNICODE_TYPE environment setting. If the driver returns any error on this call to SQLSetConnectAttr, the Driver Manager assumes that the driver does not support this connection attribute.

If an error occurs, the Driver Manager returns a warning. The Driver Manager does not convert all bound parameter data from the application Unicode type to the driver Unicode type specified by SQL_ATTR DRIVER_UNICODE_TYPE. Neither does it convert all data bound as SQL_C_WCHAR to the application Unicode type specified by SQL_ATTR_APP_UNICODE_TYPE.

Based on the information it has gathered prior to connection, the Driver Manager either does not have to convert function calls, or, before calling the driver, it converts to either UTF-8 or UTF-16 all string arguments to calls to the ODBC "W" functions.

References

The Java Tutorials, http://docs.oracle.com/javase/tutorial/i18n/index.html

Unicode Support in the Solaris Operating Environment, May 2000, Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303-4900

Code page values

The following table lists supported code page values for the IANAAppCodePage connection option. See IANAAppCodePage on page 686 for information about this attribute.

Table 139: IANAAppCodePage Values

<table>
<thead>
<tr>
<th>Value (MIBenum)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>US_ASCII</td>
</tr>
<tr>
<td>4</td>
<td>ISO_8859_1</td>
</tr>
<tr>
<td>5</td>
<td>ISO_8859_2</td>
</tr>
<tr>
<td>6</td>
<td>ISO_8859_3</td>
</tr>
<tr>
<td>7</td>
<td>ISO_8859_4</td>
</tr>
<tr>
<td>Value (MIBenum)</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>8</td>
<td>ISO_8859_5</td>
</tr>
<tr>
<td>9</td>
<td>ISO_8859_6</td>
</tr>
<tr>
<td>10</td>
<td>ISO_8859_7</td>
</tr>
<tr>
<td>11</td>
<td>ISO_8859_8</td>
</tr>
<tr>
<td>12</td>
<td>ISO_8859_9</td>
</tr>
<tr>
<td>16</td>
<td>JIS_Encoding</td>
</tr>
<tr>
<td>17</td>
<td>Shift_JIS</td>
</tr>
<tr>
<td>18</td>
<td>EUC_JP</td>
</tr>
<tr>
<td>30</td>
<td>ISO_646_IRV</td>
</tr>
<tr>
<td>36</td>
<td>KS_C_5601</td>
</tr>
<tr>
<td>37</td>
<td>ISO_2022_KR</td>
</tr>
<tr>
<td>38</td>
<td>EUC_KR</td>
</tr>
<tr>
<td>39</td>
<td>ISO_2022_JP</td>
</tr>
<tr>
<td>40</td>
<td>ISO_2022_JP_2</td>
</tr>
<tr>
<td>57</td>
<td>GB_2312_80</td>
</tr>
<tr>
<td>104</td>
<td>ISO_2022_CN</td>
</tr>
<tr>
<td>105</td>
<td>ISO_2022_CN_EXT</td>
</tr>
<tr>
<td>109</td>
<td>ISO_8859_13</td>
</tr>
<tr>
<td>110</td>
<td>ISO_8859_14</td>
</tr>
<tr>
<td>111</td>
<td>ISO_8859_15</td>
</tr>
<tr>
<td>113</td>
<td>GBK</td>
</tr>
<tr>
<td>2004</td>
<td>HP_ROMAN8</td>
</tr>
<tr>
<td>2009</td>
<td>IBM850</td>
</tr>
<tr>
<td>2010</td>
<td>IBM852</td>
</tr>
<tr>
<td>2011</td>
<td>IBM437</td>
</tr>
<tr>
<td>2013</td>
<td>IBM862</td>
</tr>
<tr>
<td>Value (MIBenum)</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>2014</td>
<td>IBM-Thai</td>
</tr>
<tr>
<td>2024</td>
<td>WINDOWS-31J</td>
</tr>
<tr>
<td>2025</td>
<td>GB2312</td>
</tr>
<tr>
<td>2026</td>
<td>Big5</td>
</tr>
<tr>
<td>2027</td>
<td>MACINTOSH</td>
</tr>
<tr>
<td>2028</td>
<td>IBM037</td>
</tr>
<tr>
<td>2029</td>
<td>IBM038</td>
</tr>
<tr>
<td>2030</td>
<td>IBM273</td>
</tr>
<tr>
<td>2033</td>
<td>IBM277</td>
</tr>
<tr>
<td>2034</td>
<td>IBM278</td>
</tr>
<tr>
<td>2035</td>
<td>IBM280</td>
</tr>
<tr>
<td>2037</td>
<td>IBM284</td>
</tr>
<tr>
<td>2038</td>
<td>IBM285</td>
</tr>
<tr>
<td>2039</td>
<td>IBM290</td>
</tr>
<tr>
<td>2040</td>
<td>IBM297</td>
</tr>
<tr>
<td>2041</td>
<td>IBM420</td>
</tr>
<tr>
<td>2043</td>
<td>IBM424</td>
</tr>
<tr>
<td>2044</td>
<td>IBM500</td>
</tr>
<tr>
<td>2045</td>
<td>IBM851</td>
</tr>
<tr>
<td>2046</td>
<td>IBM855</td>
</tr>
<tr>
<td>2047</td>
<td>IBM857</td>
</tr>
<tr>
<td>2048</td>
<td>IBM860</td>
</tr>
<tr>
<td>2049</td>
<td>IBM861</td>
</tr>
<tr>
<td>2050</td>
<td>IBM863</td>
</tr>
<tr>
<td>2051</td>
<td>IBM864</td>
</tr>
<tr>
<td>2052</td>
<td>IBM865</td>
</tr>
<tr>
<td>Value (MIBenum)</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>2053</td>
<td>IBM868</td>
</tr>
<tr>
<td>2054</td>
<td>IBM869</td>
</tr>
<tr>
<td>2055</td>
<td>IBM870</td>
</tr>
<tr>
<td>2056</td>
<td>IBM871</td>
</tr>
<tr>
<td>2062</td>
<td>IBM918</td>
</tr>
<tr>
<td>2063</td>
<td>IBM1026</td>
</tr>
<tr>
<td>2084</td>
<td>KOI8_R</td>
</tr>
<tr>
<td>2085</td>
<td>HZ_GB_2312</td>
</tr>
<tr>
<td>2086</td>
<td>IBM866</td>
</tr>
<tr>
<td>2087</td>
<td>IBM775</td>
</tr>
<tr>
<td>2089</td>
<td>IBM00858</td>
</tr>
<tr>
<td>2091</td>
<td>IBM01140</td>
</tr>
<tr>
<td>2092</td>
<td>IBM01141</td>
</tr>
<tr>
<td>2093</td>
<td>IBM01142</td>
</tr>
<tr>
<td>2094</td>
<td>IBM01143</td>
</tr>
<tr>
<td>2095</td>
<td>IBM01144</td>
</tr>
<tr>
<td>2096</td>
<td>IBM01145</td>
</tr>
<tr>
<td>2097</td>
<td>IBM01146</td>
</tr>
<tr>
<td>2098</td>
<td>IBM01147</td>
</tr>
<tr>
<td>2099</td>
<td>IBM01148</td>
</tr>
<tr>
<td>2100</td>
<td>IBM01149</td>
</tr>
<tr>
<td>2102</td>
<td>IBM1047</td>
</tr>
<tr>
<td>2250</td>
<td>WINDOWS_1250</td>
</tr>
<tr>
<td>2251</td>
<td>WINDOWS_1251</td>
</tr>
<tr>
<td>2252</td>
<td>WINDOWS_1252</td>
</tr>
<tr>
<td>2253</td>
<td>WINDOWS_1253</td>
</tr>
</tbody>
</table>
### WorkAround options

Progress DataDirect has included non-standard connection options (workarounds) for the Hybrid Data Pipeline Driver for ODBC driver that enable you to take full advantage of packaged ODBC-enabled applications requiring non-standard or extended behavior.

### WorkArounads and WorkArounads2 options

The following list includes both WorkArounads and WorkArounads2.

**warning:** Each of these options has potential side effects related to its use. An option should only be used to address the specific problem for which it was designed. For example, WorkArounads=2 causes the driver to report that database qualifiers are not supported, even when they are. As a result, applications that use qualifiers may not perform correctly when this option is enabled.

**WorkArounads=1.** Enabling this option causes the driver to return 1 instead of 0 if the value for SQL_CURSOR_COMMIT_BEHAVIOR or SQL_CURSOR_ROLLBACK_BEHAVIOR is 0. Statements are prepared again by the driver.

**WorkArounads=2.** Enabling this option causes the driver to report that database qualifiers are not supported. Some applications cannot process database qualifiers.

---

8 These values are assigned by Progress DataDirect and do not appear in http://www.iana.org/assignments/character-sets.
WorkArounds=8. Enabling this option causes the driver to return 1 instead of -1 for SQLRowCount. If an ODBC driver cannot determine the number of rows affected by an Insert, Update, or Delete statement, it may return -1 in SQLRowCount. This may cause an error in some products.

WorkArounds=16. Enabling this option causes the driver not to return an INDEX_QUALIFIER. For SQLStatistics, if an ODBC driver reports an INDEX_QUALIFIER that contains a period, some applications return a "tablename is not a valid name" error.

WorkArounds=32. Enabling this option causes the driver to re-bind columns after calling SQLEXecute for prepared statements.

WorkArounds=64. Enabling this option results in a column name of C(position where position is the ordinal position in the result set. For example, "SELECT col1, col2+col3 FROM table1" produces the column names "col1" and C2. For result columns that are expressions, SQLColAttributes/SQL_COLUMN_NAME returns an empty string. Use this option for applications that cannot process empty string column names.

WorkArounds=256. Enabling this option causes the value of SQLGetInfo/SQL_ACTIVE_CONNECTIONS to be returned as 1.

WorkArounds=512. Enabling this option prevents ROWID results. This option forces the SQLSpecialColumns function to return a unique index as returned from SQLStatistics.

WorkArounds=2048. Enabling this option causes DATABASE= instead of DB= to be returned. For some data sources, Microsoft Access performs more efficiently when the output connection string of SQILDriverConnect returns DATABASE= instead of DB=.

WorkArounds=65536. Enabling this option strips trailing zeros from decimal results, which prevents Microsoft Access from issuing an error when decimal columns containing trailing zeros are included in the unique index.

WorkArounds=131072. Enabling this option turns all occurrences of the double quote character (") into the accent grave character ('). Some applications always quote identifiers with double quotes. Double quoting can cause problems for data sources that do not return SQLGetInfo/SQL_IDENTIFIER_QUOTE_CHAR = double_quote.

WorkArounds=524288. Enabling this option forces the maximum precision/scale settings. The Microsoft Foundation Classes (MFC) bind all SQL_DECIMAL parameters with a fixed precision and scale, which can cause truncation errors.

WorkArounds=1048576. Enabling this option overrides the specified precision and sets the precision to 256. Some applications incorrectly specify a precision of 0 for character types when the value will be SQL_NULL_DATA.

WorkArounds=2097152. Enabling this option overrides the specified precision and sets the precision to 2000. Some applications incorrectly specify a precision of -1 for character types.

WorkArounds=4194304. Enabling this option converts, for PowerBuilder users, all catalog function arguments to uppercase unless they are quoted.

WorkArounds=16777216. Enabling this option allows MS Access to retrieve Unicode data types as it expects the default conversion to be to SQL_C_CHAR and not SQL_C_WCHAR.

WorkArounds=33554432. Enabling this option prevents MS Access from failing when SQLError returns an extremely long error message.

WorkArounds=67108864. Enabling this option allows parameter bindings to work correctly with MSDASQL.

WorkArounds=536870912. Enabling this option allows re-binding of parameters after calling SQLEXecute for prepared statements.

WorkArounds=1073741824. Enabling this option addresses the assumption by the application that ORDER BY columns do not have to be in the SELECT list. This assumption may be incorrect for data sources such as Informix.
**WorkArounds2=2.** Enabling this option causes the driver to ignore the ColumnSize/DecimalDigits specified by the application and use the database defaults instead. Some applications incorrectly specify the ColumnSize/DecimalDigits when binding timestamp parameters.

**WorkArounds2=4.** Enabling this option reverses the order in which Microsoft Access returns native types so that Access uses the most appropriate native type. Microsoft Access uses the last native type mapping, as returned by SQLGetTypeInfo, for a given SQL type.

**WorkArounds2=8.** Enabling this option causes the driver to add the bindoffset in the ARD to the pointers returned by SQLParamData. This is to work around an MSDASQL problem.

**WorkArounds2=16.** Enabling this option causes the driver to ignore calls to SQLFreeStmt(RESET_PARAMS) and only return success without taking other action. It also causes parameter validation not to use the bind offset when validating the charoctetlength buffer. This is to work around a MSDASQL problem.

**WorkArounds2=24.** Enabling this option allows a flat-file driver, such as dBASE, to operate properly under MSDASQL.

**WorkArounds2=32.** Enabling this option appends "DSN=“ to a connection string if it is not already included. Microsoft Access requires "DSN” to be included in a connection string.

**WorkArounds2=128.** Enabling this option causes 0 to be returned by SQLGetInfo(SQL_ACTIVE_STATEMENTS). Some applications open extra connections if SQLGetInfo(SQL_ACTIVE_STATEMENTS) does not return 0.

**WorkArounds2=256.** Enabling this option causes the driver to return Buffer Size for Long Data on calls to SQLGetData with a buffer size of 0 on columns of SQL type SQL_LONGVARCHAR or SQL_LONGVARBINARY. Applications should always set this workaround when using MSDASQL and retrieving long data.

**WorkArounds2=512.** Enabling this option causes the flat-file drivers to return old literal prefixes and suffixes for date, time, and timestamp data types. Microsoft Query 2000 does not correctly handle the ODBC escapes that are currently returned as literal prefix and literal suffix.

**WorkArounds2=1024.** Enabling this option causes the driver to return "N" for SQLGetInfo(SQL_MULT_RESULT_SETS). ADO incorrectly interprets SQLGetInfo(SQL_MULT_RESULT_SETS) to mean that the contents of the last result set returned from a stored procedure are the output parameters for the stored procedure.

**WorkArounds2=2048.** Enabling this option causes the driver to accept 2.x SQL type defines as valid. ODBC 3.x applications that use the ODBC cursor library receive errors on bindings for SQL_DATE, SQL_TIME, and SQL_TIMESTAMP columns. The cursor library incorrectly rebinds these columns with the ODBC 2.x type defines.

**WorkArounds2=4096.** Enabling this option causes the driver to internally adjust the length of empty strings. The ODBC Driver Manager incorrectly translates lengths of empty strings when a Unicode-enabled driver uses a non-Unicode driver. Use this workaround only if your application is Unicode-enabled.

**WorkArounds2=8192.** Enabling this option causes Microsoft Access not to pass the error -7748. Microsoft Access only asks for data as a two-byte SQL_C_WCHAR, which is an insufficient buffer size to store the UCS2 character and the null terminator; thus, the driver returns a warning, "01004 Data truncated" and returns a null character to Microsoft Access. Microsoft Access then passes error -7748.

---

**Using the WorkAround options**

To use these options, we recommend that you create a separate user data source for each application.

You can make the change by updating the Registry. After you create the data source,

- On Windows, using the registry editor REGEDIT, open the HKEY_CURRENT_USER\SOFTWARE\ODBC\ODBC.INI section of the registry. Select the data source that you created.
• On UNIX/Linux, using a text editor, open the odbc.ini file to edit the data source that you created.

Add the string WorkArounds= (or WorkArounds2=) with a value of $n$ (WorkArounds=$n$ or WorkArounds2=$n$),
where the value $n$ is the cumulative value of all options added together. For example, if you wanted to use both
WorkArounds=1 and WorkArounds=8, you would enter in the data source:

```
WorkArounds=9
```

**warning:** Each of these options has potential side effects related to its use. An option should only be used to
address the specific problem for which it was designed. For example, WorkArounds=2 causes the driver to
report that database qualifiers are not supported, even when they are. As a result, applications that use qualifiers
may not perform correctly when this option is enabled.
Configuring Hybrid Data Pipeline for JDBC

For details, see the following topics:

- Getting started with the JDBC driver
- Supported Features
- Using connection pooling
- Testing your application
- Troubleshooting
- Connection properties reference
- JDBC support
- DataDirect connection pooling
- JDBC extensions
- SQL escape sequences

Getting started with the JDBC driver

JDBC™ provides an API that Java applications can use to access a database using Structured Query Language (SQL). The Hybrid Data Pipeline Driver for JDBC, which is compliant with JDBC 4.0 and earlier specifications, works with the Hybrid Data Pipeline connectivity service to provide SQL access to supported data stores from any JDBC application.
The Hybrid Data Pipeline Driver for JDBC connects to a Hybrid Data Pipeline data source, which in turn, connects to the data store. The DataDirect connectivity service executes JDBC calls from the application and supports operations on data such as queries, inserts, updates, deletes, invocation of stored procedures and queries of meta data.

Once you have installed the Hybrid Data Pipeline Driver for JDBC, obtaining data with a JDBC application requires the following general steps:

1. Log in to the Hybrid Data Pipeline dashboard and create a data source. A data source defines how to connect to a data store.

2. Optionally, test the connection to the data store as described in Testing the JDBC connection to a Hybrid Data Pipeline Data Source on page 732.

3. Configure your application to connect to the Hybrid Data Pipeline Driver for JDBC data source as described in Connecting from an Application to Hybrid Data Pipeline on page 734.

As you configure the Hybrid Data Pipeline data source, the JDBC data source, and your application, you will be working with several sets of credentials and connection parameters.

As part of the JDBC URL, the application passes in the user name and password for your Hybrid Data Pipeline account. It also passes in the data source name as the Hybrid Data Pipeline Data Source. If the credentials for the data store are not saved in the data source, the application will need to supply them as part of the URL.

Testing the JDBC connection to a Hybrid Data Pipeline Data Source

Before modifying an application with the URL to connect, you can use DataDirect Test’” to verify the URL. The screen shots in this section were taken on a Windows system.

To test the connection from the driver to a data store, follow these steps:

1. Navigate to the driver installation directory. For example, to point to the file for an installation on /opt/jdbc, you navigate to:
   
   /opt/jdbc/Hybrid_for_JDBC

2. From the testforjdbc folder, run the platform-specific tool:
   
   testforjdbc.sh (on UNIX and Linux systems)
   
   The Test for JDBC Tool window appears:
3. Click **Press Here to Continue**.

The main dialog appears:

4. From the menu bar, select **Connection > Connect to DB**.

The **Select A Database** dialog appears:
5. Select the **Database** field to edit the URL.

6. At the end of the URL, replace `myDataSource` with the name of the **Data Source** defined in the Hybrid Data Pipeline dashboard.

7. If you did not store credentials for the data store in the Data Source, add these parameters to the end of the JDBC URL:
   - `datastoreUserId=<user_name>`
   - `datastorePassword=<password>` (optionally, if the account requires a security token, append it to the password)

   For example, with a user name of `me@mycompany.com`, a password of `myPassword`, and a security token of `zzzzzzzzzzzzz`, append the following to the URL: `datastoreUserId=me@mycompany.com;datastorePassword=myPasswordzzzzzzzzzzzzz`

8. For **User Name**, type your Hybrid Data Pipeline account user name.

9. For **Password**, type your Hybrid Data Pipeline account password.

10. Click **Connect**.

    The dialog reports whether the connection was successful.

Now that you have verified the connection URL, you can use that in your JDBC application, as described in [Connecting from an Application to Hybrid Data Pipeline](#) on page 734.

### Connecting from an Application to Hybrid Data Pipeline

You can use Hybrid Data Pipeline to provide access to data stores for packaged applications and custom applications. Packaged applications define their own way of specifying a JDBC connection. However, the JDBC URL for packaged applications will be identical to the URL you would use in a custom application.

Once the Hybrid Data Pipeline Driver for JDBC is installed and configured, you can connect from a custom application to a Hybrid Data Pipeline **Data Source** in either of the following ways:

- Using the JDBC Driver Manager, by specifying the connection URL in the `DriverManager.getConnection()` method. See [Connecting using the JDBC Driver Manager](#) on page 735 for more information.

- Creating a JDBC data source that can be accessed through the Java Naming Directory Interface (JNDI). See [Connecting using JDBC data sources](#) on page 736 for sample code that you can use as a template for creating and using your own JDBC data sources.
**JDBC URL**

A JDBC URL includes the following elements:

```
protocol://[hostname:port][;property=value[;...]]
```

A HybridDataPipelineDataSource property specifies the data source to which you want to connect. For example, the following example URL assumes:

- A data source name of myOraDS
- The data store credentials are stored in the myOraDS Data Source
- Encryption is not required

```
jdbc:datadirect:ddhybrid://myhost:8080;hybridDataPipelineDataSource=myOraDS;encryptionMethod=noEncryption
```

The URL elements to connect to a data source defined in the Hybrid Data Pipeline dashboard are described in the following table.

**Table 140: JDBC URL Elements**

<table>
<thead>
<tr>
<th>URL Element</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>The protocol for the Hybrid Data Pipeline connectivity service.</td>
<td><code>jdbc:datadirect:ddhybrid</code></td>
</tr>
<tr>
<td>HostName</td>
<td>The DNS name of the machine where Hybrid Data Pipeline is installed.</td>
<td><code>myhost</code></td>
</tr>
<tr>
<td>Port</td>
<td>Port that the Hybrid Data Pipeline service is listening to.</td>
<td><code>8080</code></td>
</tr>
<tr>
<td><code>property=value</code></td>
<td>Optional connection properties for the driver.</td>
<td><code>HybridDataPipelineDataSource</code> is the only required connection property. The name of the Hybrid Data Pipeline data source is saved in your Hybrid Data Pipeline account. For more information on other optional connection properties, see <a href="#">Connection Properties</a> on page 756.</td>
</tr>
</tbody>
</table>

**Connecting using the JDBC Driver Manager**

To use the JDBC Driver manager:

- The driver JAR file must be defined in the application's CLASSPATH.
- Within your application, you need to pass in the connection URL.

Follow these steps to add the driver to a CLASSPATH, register it, and add the appropriate calls in your application:

1. Set your system `CLASSPATH` to include the driver jar file as shown, where `install_dir` is the path to your product installation directory:
UNIX Example: CLASSPATH=.:~/home/user1/ddhybridjdbc/lib/ddhybrid.jar

2. Pass the connection URL in the application. The URL includes the user name and password for your Hybrid Data Pipeline connectivity service account and the name of the data source defined in the connectivity service. You must also pass in the credentials to the data store if they are not saved in the data source:

   • This example assumes that the data source contains login credentials for the data store, the data source name is myDataSource, myusername and mypassword are the login credentials for the Hybrid Data Pipeline service:

     ```java
     Connection conn = DriverManager.getConnection("jdbc:datadirect:ddhybrid://myserver:8080;hybridDataPipelineDataSource=mydatasource", user=myusername;password=mypassword;encryptionMethod=noEncryption;);
     ```

   • This example assumes that login credentials are not stored in the data source definition, the data source name and connectivity service login credentials are the same as the previous example, and the data store user ID and password are test and secret:

     ```java
     Connection conn = DriverManager.getConnection("jdbc:datadirect:ddhybrid://myserver:8080;hybridDataPipelineDataSource=mydatasource;datasourceUserId=test;datasourcePassword=secret;encryptionMethod=noEncryption;user=myusername;password=mypassword");
     ```

Other connection properties specific to the type of data store are set in the data source definition. To modify those, log in to your Hybrid Data Pipeline account.

**Connecting using JDBC data sources**

A JDBC data source is a Java object, specifically a DataSource object, that defines connection information required for a JDBC driver to connect to the database. Each JDBC driver vendor provides their own data source implementation for this purpose. Progress DataDirect provides a DataSource object for storing the connection information needed for the JDBC driver to connect to a Hybrid Data Pipeline data source, which in turn provides access to a data store.

JDBC data sources work with the Java Naming Directory Interface (JNDI) naming service, providing an extra level of abstraction that allows you to create and manage JDBC data sources (in this case, a Hybrid Data Pipeline connectivity service data source) separately from the applications that use them. The connection information is defined outside of the application, minimizing the effort to reconfigure applications when data source parameters change. The applications only refer to the name of the JDBC data source and therefore, do not need to change.

The Hybrid Data Pipeline Driver for JDBC data source class implements the following JDBC interfaces:

- javax.sql.DataSource.
- javax.sql.ConnectionPoolDataSource allows applications to use connection pooling.

To create your own JDBC data source implementation, consider the following requirements:

- If you plan to connect using a JNDI File System Service Provider, the fscontext.jar and providerutil.jar files that are shipped with the JNDI File System Service Provider, must be on your classpath. To download the JNDI File System Service Provider, go to the following Web site and select a JNDI version:

Calling a JDBC data source in an application

Applications can call a JDBC data source using a logical name to retrieve the javax.sql.DataSource object. This object loads the specified driver and can be used to establish a connection to the Hybrid Data Pipeline service.

Once the JDBC data source has been registered with JNDI, it can be used by your JDBC application as shown in the following code example, where myusername and mypassword are the credentials for your Hybrid Data Pipeline user account.

```java
Context ctx = new InitialContext();
DataSource ds = (DataSource)ctx.lookup("Employee");
Connection con = ds.getConnection("myusername","mypassword");
```

In this example, the JNDI environment is first initialized. Next, the initial naming context is used to find the logical name of the JDBC data source (Employee). The `Context.lookup()` method returns a reference to a Java object, which is narrowed to a javax.sql.DataSource object. Finally, the `DataSource.getConnection()` method is called to establish a connection.

**Note:** If the login credentials of the data store are not stored in the specified data source, the JDBC data source must include them.

Connecting Through a Proxy Server

In some environments, your application may need to use a proxy server to connect to the Hybrid Data Pipeline connectivity service.

If your application connects to the Hybrid Data Pipeline connectivity service through a proxy server, it needs to provide the following connection information:

- Server name or IP address of the proxy server (required)
- Port number on which the proxy server is listening for HTTPS requests (required)
- Credentials for the proxy server (required if the server requires authentication, consult your system administrator)

Specify the proxy server connection information in the JDBC URL or JDBC data source using the `ProxyHost`, `ProxyPort`, `ProxyUser`, and `ProxyPassword` connection properties. The following example illustrates use of these properties (URL elements are shown on separate lines for readability; enter them without line breaks):

```java
jdbc:datadirect:ddhybrid://myserver:8080;
hybridDataPipelineDataSource=myDataSource;
proxyHost=myProxyHost;
proxyPort=1234;
proxyUser=theProxyUser;
proxyPassword=myProxyPassword;
user=mycloudusername,
password=mycloudpassword
```

Driver and Data Source Classes

The driver class for the Hybrid Data Pipeline Driver for JDBC is:

com.ddtek.jdbc.ddhybrid.DDHybridDriver
Two data source classes are provided with the driver. Which data source class you use depends on the JDBC functionality your application requires. The following table shows the recommended data source class to use with different JDBC specifications.

**Table 141: Choosing the Data Source Class**

<table>
<thead>
<tr>
<th>If your application requires...</th>
<th>Choose data source class</th>
</tr>
</thead>
<tbody>
<tr>
<td>JDBC 4.0 functionality and higher</td>
<td>com.ddtek.jdbcx.ddhybrid.DDHybridDataSource40</td>
</tr>
<tr>
<td>JDBC 3.0 functionality and earlier specifications</td>
<td>com.ddtek.jdbcx.ddhybrid.DDHybridDataSource</td>
</tr>
</tbody>
</table>

See [Connecting using JDBC data sources](#) on page 736 for information about data sources.

### Supported Features

This section describes how the Hybrid Data Pipeline Driver for JDBC implements standard JDBC, security, and connectivity features.

#### Data encryption

The driver supports *Secure Sockets Layer (SSL)* data encryption. SSL is an industry-standard protocol for sending encrypted data over database connections. SSL secures the integrity of your data by encrypting information and providing client/server authentication.

Communication between the Hybrid Data Pipeline Driver for JDBC and the Hybrid Data Pipeline connectivity service, including user IDs and passwords, can be encrypted using SSL.

#### Unicode

Multilingual JDBC applications can be developed on any operating system using the driver to access both Unicode and non-Unicode enabled data stores. Internally, Java applications use UTF-16 Unicode encoding for string data. When fetching data, the driver automatically performs the conversion from the character encoding used by the data stores to UTF-16. Similarly, when inserting or updating data in the data stores, the driver automatically converts UTF-16 encoding to the character encoding used by the data store.

The JDBC API provides mechanisms for retrieving and storing character data encoded as Unicode (UTF-16) or ASCII. Additionally, the Java String object contains methods for converting UTF-16 encoding of string data to or from many popular character encodings.

#### Scrollable cursors

The driver supports scroll-insensitive result sets and updatable result sets.

**Note:** When the driver cannot support the requested result set type or concurrency, it automatically downgrades the cursor and generates one or more SQLWarnings with detailed information.
Large objects (LOBs)

The driver allows you to retrieve and update long data, specifically LONGVARBINARY and LONGVARCHAR data, using JDBC methods designed for Blobs and Clobss. When using these methods to update long data as Blobs or Clobss, the updates are made to the local copy of the data contained in the Blob or Clob object.

Retrieving and updating long data using JDBC methods designed for Blobs and Clobss provides some of the same benefits as retrieving and updating Blobs and Clobss, such as:

- Provides random access to data
- Allows searching for patterns in the data, such as retrieving long data that begins with a specific character string

To provide these benefits normally associated with Blobs and Clobss, data must be cached. Because data is cached, your application will incur a performance penalty, particularly if data is read only once sequentially. This performance penalty can be severe if the size of the long data is larger than available memory.

Rowsets

The driver supports any JSR 114 implementation of the RowSet interface, including:

- CachedRowSets
- FilteredRowSets
- WebRowSets
- JoinRowSets
- JDBCRowSets


Auto-generated keys

The driver supports retrieving the values of auto-generated keys. An auto-generated key returned by the driver is the value of an auto-increment column.

An application can return values of auto-generated keys when it executes an Insert statement. How you obtain these values depends on whether you are using an Insert statement that contains parameters.

- When using an Insert statement that contains no parameters, the driver supports the following forms of the Statement.executeUpdate methods to inform the driver to return the values of auto-generated keys:
  - Statement.executeUpdate(String sql, int autoGeneratedKeys)
  - Statement.executeUpdate(String sql, int[] columnIndexes)
  - Statement.executeUpdate(String sql, String[] columnNames)
  - Statement.executeUpdate(String sql, int autoGeneratedKeys)
  - Statement.executeUpdate(String sql, int[] columnIndexes)
  - Statement.executeUpdate(String sql, String[] columnNames)
• When inserting data using a prepared statement, the driver supports the following forms of the `Connection.prepareStatement` method to inform the driver to return the values of auto-generated keys:
  
  • `Connection.prepareStatement(String sql, int autoGeneratedKeys)`
  • `Connection.prepareStatement(String sql, int[] columnIndexes)`
  • `Connection.prepareStatement(String sql, String[] columnNames)`

  An application can retrieve values of auto-generated keys using the `Statement.getGeneratedKeys()` method. This method returns a `ResultSet` object with a column for each auto-generated key.

**Using IP addresses**

The driver supports Internet Protocol (IP) addresses in IPv4 and IPv6 format. To connect to IPv6 addresses, the client or application server requires J2SE 5.0 or higher.

If your network supports named servers, the server name specified in the connection URL or data source can resolve to an IPv4 or IPv6 address. For example, the server name `HybridServer` in the following URL can resolve to either type of address:

```
jdbc:datadirect:ddhybrid://myserver:8080;hybridDataPipelineDataSource=mydatasource;
  encryptionMethod=noEncryption;user=myusername;password=mypassword
```

Alternatively, you can specify addresses using IPv4 or IPv6 format in the server name portion of the connection URL. For example, the following connection URL specifies the server using IPv4 format:

```
jdbc:datadirect:ddhybrid://123.456.78.90:8080;hybridDataPipelineDataSource=mydatasource;
  encryptionMethod=noEncryption;user=myusername;password=mypassword
```

You also can specify addresses in either format using the `ServerName` data source property. The following example shows a data source definition that specifies the server name using IPv6 format:

```
jdbc:datadirect:ddhybrid://[2001:DB8:0:0:8:800:200C:417A]:8080;
  hybridDataPipelineDataSource=mydatasource; encryptionMethod=noEncryption;
  user=myusername;password=mypassword;...
```

**Note:** When specifying IPv6 addresses in a connection URL or data source property, the address must be enclosed by brackets.

In addition to the normal IPv6 format, the drivers support IPv6 alternative formats for compressed and IPv4/IPv6 combination addresses. For example, the following connection URL specifies the server using IPv6 format, but uses the compressed syntax for strings of zero bits:

```
jdbc:datadirect:ddhybrid://[2001:DB8:0:0:8:800:200C:417A]:50000;
  DDDHybridDataSource=jdbc;User=test;Password=secret
```

Similarly, the following connection URL specifies the server using a combination of IPv4 and IPv6:

```
jdbc:datadirect:ddhybrid://[0000:0000:0000:0000:0000:FFFF:123.456.78.90]:8080;
  hybridDataPipelineDataSource=mydatasource; encryptionMethod=noEncryption;
  user=myusername; password=mypassword
```

For complete information about IPv6, go to the following URL:

```
http://tools.ietf.org/html/rfc4291#section-2.2
```
SQL support

The Hybrid Data Pipeline Driver for JDBC, working in conjunction with the Hybrid Data Pipeline connectivity service, supports standard SQL 92. Specific support is determined by the data store to which the Hybrid Data Pipeline connectivity service is connected. For example, the SQL supported by Salesforce is different than the SQL supported by Oracle.

Using connection pooling

Typically, connection creation is the most expensive operation that an application performs. Connection pooling allows you to reuse connections rather than create a new one every time an application requires a connection to the Hybrid Data Pipeline connectivity service.

See DataDirect connection pooling on page 815 for reference information on the interfaces and methods.

How connection pooling works

Connection pooling shares connections across different user requests to maintain performance and reduce the number of new connections that must be created. Compare the following transaction sequences to picture the efficiency offered by pooling connections.

Example A: Without connection pooling

1. The application creates a connection.
2. The application sends a query to the Hybrid Data Pipeline connectivity service.
3. The application obtains query results.
4. The application displays the result to the end user.
5. The application ends the connection.

Example B: With connection pooling

1. The application requests a connection from the connection pool.
2. If an unused connection exists, it is returned by the pool; otherwise, the pool creates a new connection.
3. The application sends a query to the Hybrid Data Pipeline connectivity service.
4. The application obtains query results.
5. The application displays the result to the end user.
6. The application closes the connection, which returns the connection to the pool.

Note: The application calls the close() method, which allows the connection to remain open. The pool receives notification of the close request.
Connection pooling is performed in the background and does not affect how an application is coded. To use connection pooling, an application must use a `DataSource` object (an object implementing the `DataSource` interface) to obtain a connection instead of using the `DriverManager` class. A `DataSource` object registers with a JNDI naming service. Once a `DataSource` object is registered, the application retrieves it from the JNDI naming service in the standard way.

**Note:** A class implementing the `DataSource` interface may or may not provide connection pooling.

There is a one-to-one relationship between a JDBC connection pool and a Hybrid Data Pipeline Driver for JDBC data source, so the number of connection pools used by an application depends on the number of data sources configured to use connection pooling. If multiple applications are configured to use the same data source, those applications share the same connection pool as shown in the following figure.

![Connection Pool Diagram](image)

An application may use only one data source, but allow multiple users, each with their own set of login credentials. The connection pool contains connections for all unique users using the same data source as shown in the following figure.

![Connection Pool Diagram](image)

A connection pool contains two types of connections:

- **Active connection** is a connection that is in use by the application.
- **Idle connection** is a connection in the connection pool that is available for use.

Connection pool implementations, such as the DataDirect Connection Pool Manager, use objects that implement the `javax.sql.ConnectionPoolDataSource` interface to create the connections managed in the pool. All Progress DataDirect `DataSource` objects implement the `ConnectionPoolDataSource` interface.

You can create your own connection pool implementation using the DataDirect Connection Pool Manager as described in **Using a DataDirect connection pool** on page 743. A connection pool implementation creates `PooledConnections`, using the `getPooledConnection()` method of the `ConnectionPoolDataSource` interface. Then, the Pool Manager registers itself as a listener to the `PooledConnection`. When an application requests a connection, the Pool Manager assigns an available connection. If a connection is unavailable, the Pool Manager establishes a new connection and assigns it to that application.

When the application closes the connection, the Pool Manager is notified by the driver by the `ConnectionEventListener` interface that the connection is free and available for reuse. The Pool Manager is also notified by the `ConnectionEventListener` interface if the connection is corrupted so that the Pool Manager can remove that connection from the pool.
Using a DataDirect connection pool

To use DataDirect Connection Pooling, perform these steps:

1. Create and register with JNDI a Hybrid Data Pipeline Driver for JDBC DataSource object. Once created, this DataSource object can be used by a connection pool (PooledConnectionDataSource object created in Step 2 on page 743) to create connections for one or multiple connection pools.

2. To create a connection pool, you must create and register with JNDI a PooledConnectionDataSource object. A PooledConnectionDataSource creates and manages one or multiple connection pools. The PooledConnectionDataSource uses the driver DataSource object created in Step 1 on page 743 to create the connections for the connection pool.

Creating a Driver DataSource object

The following Java code example creates a Hybrid Data Pipeline Driver for JDBC DataSource object and registers it with a JNDI naming service.

Note: The DataSource class implements the ConnectionPoolDataSource interface for pooling in addition to the DataSource interface for non-pooling.

```java
import com.ddtek.jdbcx.ddhybrid.DDHybridDataSource;
import javax.sql.*;
import java.sql.*;
import javax.naming.*;
import javax.naming.directory.*;
import java.util.Hashtable;

public class OracleDataSourceRegisterJNDI
{
    public static void main(String argv[])
    {
        try {
            // Set up data source reference data for naming context:
            // ----------------------------------------------------
            // Create a class instance that implements the interface
            // ConnectionPoolDataSource
            DDHybridDataSource ds = new DDHybridDataSource();
            ds.setDescription("Hybrid Data Pipeline on Sparky - Data Source");
        }
    }
}
```
Creating the DataDirect connection pool

The following Java code creates a PooledConnectionDataSource object and registers it with a JNDI naming service.

To specify the driver DataSource object to be used by the connection pool to create pooled connections, set the parameter of the DataSourceName() method to the JNDI name of a registered driver DataSource object. For example, the following code sets the parameter of the DataSourceName method to the JNDI name of the driver DataSource object created in Creating a Driver DataSource object on page 743.

The PooledConnectionDataSource class is provided by the DataDirect com.ddtek.pool package. See PooledConnectionDataSource on page 815 for a description of the methods supported by the PooledConnectionDataSource class.

// This code creates a data source and registers it to a // JNDI naming service. // This data source uses the PooledConnectionDataSource // implementation provided by the DataDirect com.ddtek.pool package. // This data source refers to a registered // DataDirect Hybrid Data Pipeline Driver for JDBC DataSource object. // This data source registers its name as <jdbc/PoolHybridSparky>. // NOTE: To connect using a data source, the driver needs to access // a JNDI data store to persist the data source information. // To download the JNDI File System Service Provider, go to: // http://www.oracle.com/technetwork/java/javasebusiness/downloads/ // java-archive-downloads-java-plat-419418.html#7110-jndi-1.2.1-oth-JPR// Make sure that the fscontext.jar and providerutil.jar files from the // download are on your classpath.

// From the DataDirect connection pooling package:
import com.ddtek.pool.PooledConnectionDataSource;
import javax.sql.*;
import java.sql.*;
import java.util.Hashtable;
public class PoolMgrDataSourceRegisterJNDI
{   public static void main(String argv[])
{   

try {
    // Set up data source reference data for naming context:
    // Create a pooling manager's class instance that implements
    // the interface DataSource
    PooledConnectionDataSource ds = new PooledConnectionDataSource();
    ds.setDescription("Sparky Hybrid Pipeline - Data Source");
    // Specify a registered driver DataSource object to be used
    // by this data source to create pooled connections
    ds.setDataSourceName("jdbc/HybridSparky");
    // The pool manager will be initiated with 5 physical connections
    ds.setInitialPoolSize(5);
    // The pool maintenance thread will make sure that there are 5
    // physical connections available
    ds.setMinPoolSize(5);
    // The pool maintenance thread will check that there are no more
    // than 10 physical connections available
    ds.setMaxPoolSize(10);
    // The pool maintenance thread will wake up and check the pool
    // every 20 seconds
    ds.setPropertyCycle(20);
    // The pool maintenance thread will remove physical connections
    // that are inactive for more than 300 seconds
    ds.setMaxIdleTime(300);
    // Set tracing off because we choose not to see an output listing
    // of activities on a connection
    ds.setTracing(false);
    // Set up environment for creating initial context
    Hashtable env = new Hashtable();
    env.put(Context.INITIAL_CONTEXT_FACTORY,
            "com.sun.jndi.fscontext.RefFSContextFactory");
    env.put(Context.PROVIDER_URL, "file:c:\JDBCDataSource");
    Context ctx = new InitialContext(env);
    // Register this data source to the JNDI naming service
    ctx.bind("jdbc/PoolHybridSparky", ds);
    catch (Exception e) {
        System.out.println(e);
        return;
    }
}

Connecting to a JDBC Data Source using a connection pool

Once a connection pool has been created and registered with JNDI, it can be used by your JDBC application
when it creates the connection to the JDBC data source as shown in the following code snippet, typically
through a third-party connection pool tool:

    Context ctx = new InitialContext();
    DataSource ds = (DataSource) ctx.lookup("jdbc/PoolHybridSparky");
    Connection conn = ds.getConnection("DDusername", "DDpassword");

In this example, first, the JNDI environment is initialized. Next, the initial naming context is used to find the data
source associated with the connection pool defined in the previous section using the logical name of that pool
(jdbc/PoolHybridSparky). The Context.lookup method returns a reference to a Java object, which is narrowed
to a javax.sql.PooledDataSource object. Next, the PooledDataSource.getConnection() method is
called to establish a connection with the JDBC data source.
Closing the DataDirect connection pool

The DataDirect Connection Pool Manager is notified automatically when an application stops running. Use the `PooledConnectionDataSource.close()` method to explicitly close the pool while the application is running. For example, if changes are made to the pool configuration using a pool management tool, the `PooledConnectionDataSource.close()` method can be used to force the connection pool to close and re-create the pool using the new configuration values.

Complete example of using a connection pool

The following example shows Java code that looks up and uses the JNDI-registered DataDirect connection pool's `PooledConnectionDataSource` object. Creating the DataDirect connection pool on page 744 provides the Java code for creating and registering the `PooledConnectionDataSource` object.

```java
//********************************************************************
// Test program to look up and use a JNDI-registered data source.
// To run the program, specify the JNDI lookup name for the
// command-line argument, for example:
// java TestDataSourceApp <jdbc/HybridSparky>
//********************************************************************
import javax.sql.*;
import java.sql.*;
import javax.naming.*;
import java.util.Hashtable;
public class TestDataSourceApp {
    public static void main(String argv[]) {
        String strJNDILookupName = "";
        // Get the JNDI lookup name for a data source
        int nArgv = argv.length;
        if (nArgv != 1) {
            // User does not specify a JNDI lookup name for a data source,
            System.out.println("Please specify a JNDI name for your data source");
            System.exit(0);
        } else {
            strJNDILookupName = argv[0];
        }
        DataSource ds = null;
        Connection con = null;
        Context ctx = null;
        Hashtable env = null;
        long nStartTime, nStopTime, nElapsedTime;
        // Set up environment for creating InitialContext object
        env = new Hashtable();
        env.put(Context.INITIAL_CONTEXT_FACTORY,
            "com.sun.jndi.fscontext.RefFSContextFactory");
        env.put(Context.PROVIDER_URL, "file:c:\\JDBCDataSource");
        try {
            // Retrieve the DataSource object that is bound to the logical
            // lookup JNDI name
            ctx = new InitialContext(env);
            ds = (DataSource) ctx.lookup(strJNDILookupName);
            catch (NamingException eName) {
                System.out.println("Error looking up "+strJNDILookupName + ": " + eName);
                System.exit(0);
            }
            int numOfTest = 4;
            int [] nCount = {100, 100, 1000, 3000};
            for (int i = 0; i < numOfTest; i ++) {
                // Log the start time
```
Testing your application

In addition to testing your connection to the Hybrid Data Pipeline service, you can use DataDirect Test™ to test your JDBC connection. DataDirect Test contains menu selections that correspond to specific JDBC functions, for example, connecting with the driver to the Hybrid Data Pipeline service or passing a SQL statement. DataDirect Test allows you to perform the following tasks:

- Execute a single JDBC method or execute multiple JDBC methods simultaneously, so that you can easily perform some common tasks, such as returning result sets
- Display the results of all JDBC function calls in one window, while displaying fully commented, JDBC code in an alternate window

Configuring DataDirect Test

The default DataDirect Test configuration file is:

```
install_dir/testfordjdbc/Config.txt
```

where:

```
install_dir
```

is your product installation directory.
You can edit the DataDirect Test configuration file for your environment using a text editor. All parameters are configurable, but the most commonly configured parameters are listed here:

**Drivers**

is a list of colon-separated JDBC driver classes.

**DefaultDriver**

is com.ddtek.jdbc.ddhybrid.DDHybridDriver.

**Databases**

is a list of comma-separated JDBC URLs. The first item in the list appears as the default in the database selection window. You can use one of these URLs as a template when you make a JDBC connection.

**InitialContextFactory**

is com.sun.jndi.fscontext.RefFSContextFactory if you are using file system data sources, or com.sun.jndi.ldap.LdapCtxFactory if you are using LDAP.

**ContextProviderURL**

depends on whether you are using file system data sources or using LDAP. If you are using file system data sources, specify the location of the .bindings file. If you are using LDAP, specify your LDAP Provider URL.

**Datasources**

is a list of comma-separated JDBC data sources. The first item in the list appears as the default in the data source selection window.

To connect using a data source, DataDirect Test needs to access a JNDI data store to persist the data source information. By default, DataDirect Test is configured to use the JNDI File System Service Provider to persist the data source. To download the JNDI File System Service Provider, go to:

http://www.oracle.com/technetwork/java/javasebusiness/downloads/
java-archive-downloads-java-plat-419418.html#7110-jndi-1.2.1-oth-JPR

Make sure that the fscontext.jar and providerutil.jar files from the download are on your classpath.

**Troubleshooting**

This section discusses performance and troubleshooting.

**Installation issues**

If the installer successfully creates the product installation directory, the installer writes a file named Progress_DataDirect_Hybrid_Driver_for_JDBC.log in the install_dir/install/logs directory. Examine the log file for a record of any problems that may have occurred during the installation.
If a product installation fails completely, the installer does not create the product installation directory and writes a `Progress_DataDirect_Hybrid_Driver_for_JDBCFailed.txt` file in the machine's default temporary directory.

If you need help interpreting the contents of these files, contact Progress DataDirect technical support.

**SQL errors**

Hybrid Data Pipeline reports errors to the calling application by returning SQL exceptions. The message indicates which Hybrid Data Pipeline component generated the error.

The following components can report errors:

- **Hybrid Data Pipeline Driver for JDBC** - [DataDirect][JDBC Hybrid Driver] Example: [DataDirect][JDBC Hybrid Driver][Service]Object has been closed

- **Hybrid Data Pipeline connectivity service** - [DataDirect][JDBC Hybrid Driver][Service] Example: [DataDirect][JDBC Hybrid Driver][Service]Invalid user ID or password.

- **Hybrid Data Pipeline data store** - [DataDirect][JDBC Hybrid Driver][data_store] Example: [DataDirect][JDBC Hybrid Driver][Salesforce]Column not found: FOO in statement [SELECT foo FROM Account].

You may need to check the last JDBC call your application made and refer to the JDBC specification for the recommended action.

When a JDBC call fails it throws a SQLException. Calling `getMessage` or `toString` on the SQLException will return these messages. For example:

```java
try {
    rs = stmt.executeQuery("SELECT * FROM foobar");
} catch (SQLException e) {
    System.out.println(e.toString());
    System.out.println(e.getMessage());
}
```

**Troubleshooting an application by logging**

The driver provides flexible and comprehensive logging through Java logging. You can incorporate the driver logging with application logging or enable and configure it independently from an application. Logging can be instrumental in investigating and diagnosing issues. It also provides valuable insight into the type and number of operations requested by the application from the Hybrid Data Pipeline Driver for JDBC and requested by the driver from the Data Source. Such information can help you tune and optimize your application.

The JVM Java Logging API allows applications or components to define one or more named loggers. The Hybrid Data Pipeline Driver for JDBC supports use of this logging API for messages that pertain to the Driver.

Each logger used by the driver can be configured independently. The configuration for a logger includes what level of log messages are written, the location to which they are written, and the format of the log message.

**Configuring logging**

You can configure logging using a standard Java properties file in either of the following ways:

- Using the properties file that is shipped with your JVM.
• Using the driver.

Using the driver

By default, the driver looks for the file named `ddhybridlogging.properties` in the current working directory to load for all connections.

If a properties file is specified for the `LogConfigFile` connection property, the driver uses the following process to determine which file to load:

1. The driver looks for the file specified by the `LogConfigFile` property.
2. If the driver cannot find the file in Step 1, it looks for a properties file named `ddhybridlogging.properties` in the current working directory.
3. If the driver cannot find the file in Step 2, it abandons its attempt to load a properties file.

If any of these files exist, but the logging initialization fails for some reason while using that file, the driver writes a warning to the standard output (`System.out`), specifying the name of the properties file being used.

A sample properties file is installed in the `install_dir/testforjdbc` directory, where `install_dir` is your product installation directory. The file is named `ddhybridlogging.properties`. You can copy this file to the current working directory of your application, and modify it for your needs using a text editor.

Using the JVM

If you want to configure logging using the properties file that is shipped with your JVM, use a text editor to modify the properties file in your JVM. Typically, this file is named `logging.properties` and is located in the `JRE/lib` subdirectory of your JVM. The JRE looks for this file when it is loading.

You can also specify which properties file to use by setting the `java.util.logging.config.file` system property. At a command prompt, enter:

```java
java -Djava.util.logging.config.file=properties_file
```

where `properties_file` is the name of the properties file you want to load.

Logging Levels

Messages written to the loggers can be given different levels of importance. For example, errors that occur in the Hybrid Data Pipeline Driver for JDBC are written to a logger at the `CONFIG` level, while progress or flow information will be written to a logger at the `FINE` or `FINER` level.

The Java Logging API defines the following levels:

• SEVERE
• WARNING
• INFO
• CONFIG
• FINE
• FINER
• FINEST
Note: Log messages logged by the driver only use the CONFIG, FINE, and FINEST logging levels.

Setting the log threshold of a logger to a particular level causes the logger to write log messages of that level and higher to the log. For example, if the threshold is set to FINE, the logger writes messages of levels FINE, CONFIG, INFO, WARNING, and SEVERE to its log; messages of level FINER or FINEST are not written to the log.

The Hybrid Data Pipeline Driver for JDBC exposes loggers for the following functional areas:

- JDBC API
- Driver

**JDBC API Logger**

**Name**
com.ddtek.jdbc.spy

**Purpose**
Logs the JDBC calls made by the application to the driver and the responses from the driver back to the application.

**Message Levels**

- **FINER** - Calls to the JDBC methods are logged at the FINER level. The value of all input parameters passed to these methods and the return values passed from them are also logged, except that input parameter or result data contained in InputStream, Reader, Blob, or Clob objects are not written at this level.
- **FINEST** - In addition to the same information logged by the FINER level, input parameter values and return values contained in InputStream, Reader, Blob and Clob objects are written at this level.
- **OFF** - Calls to the JDBC methods are not logged.

**Driver Logger**

**Name**
com.ddtek.jdbcx.ddhybrid

**Purpose**
Logs the calls the driver makes to the Hybrid Data Pipeline Data Source and the responses it receives.

**Message Levels**

- **CONFIG** - Any errors or warnings detected by the driver are written at this level.
- **FINE** - In addition to the same information logged by the CONFIG level, information about calls made by the driver and responses received by the driver are written at this level. In particular, the driver calls made to execute the query and the calls to fetch or send the data are logged. The log entries for the calls to execute the query include the specific query being executed. The actual data sent or fetched is not written at this level.
- **FINEST** - In addition to the same information logged by the CONFIG and FINE levels, data associated with the calls made by the driver is written.
Troubleshooting Connection Pooling

Connection pooling allows connections to be reused rather than created each time a connection is requested. If your application is using connection pooling through the DataDirect Connection Pool Manager, you can generate a trace file that shows all the actions taken by the Pool Manager. See Using connection pooling on page 741 for information about using the Pool Manager.

Enabling Pool Manager Tracing

You can enable Pool Manager logging by calling setTracing(true) on the PooledConnectionDataSource connection. To disable tracing, call setTracing(false).

By default, the DataDirect Connection Pool Manager logs its pool activities to the standard output System.out. You can change where the Pool Manager trace information is written by calling the setLogWriter() method on the PooledConnectionDataSource connection.

Example: Pool Manager trace file

The following example shows a DataDirect Connection Pool Manager trace file. Notes provide explanations for the referenced text to help you understand the content of your own Pool Manager trace files.

The parameters with which the connection pool was created display on the *** ConnectionPool Created line and include the following:

- JNDI name used to look up the connection pool: jdbc/SparkyPool
- DataSource class associated with the connection pool:
  com.ddtek.jdbc.ddhybrid.DDHybridDataSource
- Initial pool size (number of connections created upon initialization): 5
- Min pool size (number of connections to be kept open): 5
- Max pool size (maximum number of connections at any one time): 10
- User establishing the connection: DDUser

jdbc/SparkyPool:
*** ConnectionPool Created (jdbc/SparkyPool,
   com.ddtek.jdbc.ddhybrid.DDHybridDataSource@1835282, 5, 5, 10, DDUser)
jdbc/SparkyPool: Number pooled connections = 0.
jdbc/SparkyPool: Number free connections = 0.
jdbc/SparkyPool: Enforced minimum! 9
NrFreeConnections was: 0
jdbc/SparkyPool: Number pooled connections = 5.
jdbc/SparkyPool: Number free connections = 5.
jdbc/SparkyPool: Reused free connection. 10
jdbc/SparkyPool: Number pooled connections = 5.
jdbc/SparkyPool: Number free connections = 4.
jdbc/SparkyPool: Reused free connection.
jdbc/SparkyPool: Number pooled connections = 5.
jdbc/SparkyPool: Number free connections = 3.

9 The Pool Manager checks the pool size. Because the minimum pool size is five connections, the Pool Manager creates new connections to satisfy the minimum pool size.
10 The driver requests a connection from the connection pool. The driver retrieves an available connection.
jdbc/HybridSparkyPool: Reused free connection.
jdbc/HybridSparkyPool: Number pooled connections = 5.
jdbc/HybridSparkyPool: Number free connections = 2.

jdbc/HybridSparkyPool: Reused free connection.
jdbc/HybridSparkyPool: Number pooled connections = 5.
jdbc/HybridSparkyPool: Number free connections = 1.

jdbc/HybridSparkyPool: Created new connection.  \(^{11}\)
jdbc/HybridSparkyPool: Number pooled connections = 6.
jdbc/HybridSparkyPool: Number free connections = 0.

jdbc/HybridSparkyPool: Created new connection.  \(^{11}\)
jdbc/HybridSparkyPool: Number pooled connections = 7.
jdbc/HybridSparkyPool: Number free connections = 0.

 jdbc/HybridSparkyPool: Created new connection.
jdbc/HybridSparkyPool: Number pooled connections = 8.
jdbc/HybridSparkyPool: Number free connections = 0.

jdbc/HybridSparkyPool: Created new connection.
jdbc/HybridSparkyPool: Number pooled connections = 9.
jdbc/HybridSparkyPool: Number free connections = 0.

jdbc/HybridSparkyPool: Created new connection.
jdbc/HybridSparkyPool: Number pooled connections = 10.
jdbc/HybridSparkyPool: Number free connections = 0.

jdbc/HybridSparkyPool: Created new connection.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 0.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.  \(^{12}\)
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 1.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 2.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 3.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 4.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 5.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 6.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 7.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.

\(^{11}\) The driver requests a connection from the connection pool. Because a connection is unavailable, the Pool Manager creates a new connection for the request.

\(^{12}\) A connection is closed by the application and returned to the connection pool.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 8.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 9.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 10.

jdbc/HybridSparkyPool: Connection was closed and added to the cache.
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 11.

jdbc/HybridSparkyPool: Enforced minimum! 13
NrFreeConnections was: 11
jdbc/HybridSparkyPool: Number pooled connections = 11.
jdbc/HybridSparkyPool: Number free connections = 11.

jdbc/HybridSparkyPool: Enforced maximum! 14
NrFreeConnections was: 11
jdbc/HybridSparkyPool: Number pooled connections = 10.
jdbc/HybridSparkyPool: Number free connections = 10.

jdbc/HybridSparkyPool: Enforced minimum!
NrFreeConnections was: 10
jdbc/HybridSparkyPool: Number pooled connections = 10.
jdbc/HybridSparkyPool: Number free connections = 10.

jdbc/HybridSparkyPool: Enforced maximum!
NrFreeConnections was: 10
jdbc/HybridSparkyPool: Number pooled connections = 10.
jdbc/HybridSparkyPool: Number free connections = 10.

jdbc/HybridSparkyPool: Enforced minimum!
NrFreeConnections was: 10
jdbc/HybridSparkyPool: Number pooled connections = 10.
jdbc/HybridSparkyPool: Number free connections = 10.

jdbc/HybridSparkyPool: Enforced maximum!
NrFreeConnections was: 10
jdbc/HybridSparkyPool: Number pooled connections = 10.
jdbc/HybridSparkyPool: Number free connections = 10.

jdbc/HybridSparkyPool: Dumped free connection. 15
jdbc/HybridSparkyPool: Number pooled connections = 9.
jdbc/HybridSparkyPool: Number free connections = 9.

jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 8.
jdbc/HybridSparkyPool: Number free connections = 8.

jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 7.
jdbc/HybridSparkyPool: Number free connections = 7.

jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 6.
jdbc/HybridSparkyPool: Number free connections = 6.

jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 5.

---

13 The Pool Manager checks the pool size. Because the number of connections in the connection pool is greater than the minimum pool size, five connections, no action is taken by the Pool Manager.

14 The Pool Manager checks the pool size. Because the number of connections in the connection pool is greater than the maximum pool size, 10 connections, a connection is closed and discarded from the pool.

15 The Pool Manager detects that a connection was idle in the connection pool longer than the maximum idle timeout. The idle connection is closed and discarded from the pool.
jdbc/HybridSparkyPool: Number free connections = 5.
jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 4.
jdbc/HybridSparkyPool: Number free connections = 4.
jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 3.
jdbc/HybridSparkyPool: Number free connections = 3.
jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 2.
jdbc/HybridSparkyPool: Number free connections = 2.
jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 1.
jdbc/HybridSparkyPool: Number free connections = 1.
jdbc/HybridSparkyPool: Dumped free connection.
jdbc/HybridSparkyPool: Number pooled connections = 0.
jdbc/HybridSparkyPool: Number free connections = 0.
jdbc/HybridSparkyPool: Enforced minimum! \(^{16}\) 
NrFreeConnections was: 0
jdbc/HybridSparkyPool: Number pooled connections = 5.
jdbc/HybridSparkyPool: Number free connections = 5.
jdbc/HybridSparkyPool: Enforced maximum! 
NrFreeConnections was: 5
jdbc/HybridSparkyPool: Number pooled connections = 5.
jdbc/HybridSparkyPool: Number free connections = 5.
jdbc/HybridSparkyPool: Closing a pool of the group 
jdbc/HybridSparkyPool: Pool closed \(^{18}\)
jdbc/HybridSparkyPool: Number pooled connections = 0.
jdbc/HybridSparkyPool: Number free connections = 0.

Connection properties reference

JDBC connection properties can be used with either the JDBC Driver Manager or JDBC data sources. In addition to providing the information needed to make a connection to a specific data store, the connection properties allow you to specify the characteristics of the connection, such as the number of times the driver attempts to connect to the server.

---

\(^{16}\) The Pool Manager detects that the number of connections dropped below the limit set by the minimum pool size, five connections. The Pool Manager creates new connections to satisfy the minimum pool size.

\(^{17}\) The Pool Manager closes one of the connection pools in the pool group. A pool group is a collection of pools created from the same PooledConnectionDataSource call. Different pools are created when different user IDs are used to retrieve connections from the pool. A pool group is created for each user ID that requests a connection. In our example, because only one user ID was used, only one pool group is closed.

\(^{18}\) The Pool Manager closed all the pools in the pool group. The connection pool is closed.
Connection Properties

This section lists the connection properties supported by the driver for Hybrid Data Pipeline data sources and describes each property. The properties have the form:

\[ \text{property=value} \]

You can use these connection properties with either the JDBC Driver Manager or JDBC data sources unless otherwise noted.

**Note:** All connection property names are case-insensitive. For example, Password is the same as password. Required properties are noted as such.

**Note:** The data type listed for each connection property is the Java data type used for the property value in a JDBC data source.

The following table provides a summary of the connection properties supported by the driver and their default values.

**Table 142: Driver Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvertNull on page 757</td>
<td>1 (data type check is performed if column value is null)</td>
</tr>
<tr>
<td>DataSourcePassword on page 758</td>
<td>empty string</td>
</tr>
<tr>
<td>DataSourceUserID on page 758</td>
<td>empty string</td>
</tr>
<tr>
<td>EnableCancelTimeout on page 759</td>
<td>false</td>
</tr>
<tr>
<td>EncryptionMethod</td>
<td>SSL</td>
</tr>
<tr>
<td>HostNameInCertificate</td>
<td>None</td>
</tr>
<tr>
<td>HybridDataPipelineDataSource on page 761</td>
<td>empty string</td>
</tr>
<tr>
<td>InsensitiveResultSetBufferSize on page 762</td>
<td>2048 (KB of memory)</td>
</tr>
<tr>
<td>JavaDoubleToString on page 763</td>
<td>false</td>
</tr>
<tr>
<td>LogConfigFile on page 763</td>
<td>ddhybridlogging.properties</td>
</tr>
<tr>
<td>LoginTimeout on page 764</td>
<td>0 (no timeout)</td>
</tr>
<tr>
<td>Password on page 764</td>
<td>None</td>
</tr>
<tr>
<td>ProxyHost on page 765</td>
<td>empty string</td>
</tr>
<tr>
<td>ProxyPassword on page 765</td>
<td>empty string</td>
</tr>
<tr>
<td>ProxyPort on page 766</td>
<td>empty string</td>
</tr>
</tbody>
</table>
### ConvertNull

**Purpose**
Controls how the driver handles data conversions for null values.

**Valid Values**

0 | 1

**Behavior**
If set to 0, the driver does not perform the data type check if the value of the column is null. This allows null values to be returned even though a conversion between the requested type and the column type is undefined.

If set to 1, the driver checks the data type being requested against the data type of the table column that stores the data. If a conversion between the requested type and column type is not defined, the driver generates an "unsupported data conversion" exception regardless of whether the column value is NULL.

**Default**
1

**Data Type**
int
**DataSourcePassword**

**Purpose**
Specifies the case-sensitive password that is required for logging into a backend data store, such as SQL Server or Salesforce. For web service data stores such as Salesforce, a security token may be required by the data store instance.

**Valid Values**

`password | password+securitytoken`

where:

`password`

is the password required for logging into the data store.

`password+securitytoken`

is the password required for logging into the data store plus a valid security token.

**Notes**

- The data store user ID and password may be stored in the Hybrid Data Pipeline data source definition. If that is true and you specify the user ID and password using the DataSourceUserID and DataSourcePassword connection properties, the values specified in these connection properties take precedence.

- When the data store requires a security token but it has not been stored in the Hybrid Data Pipeline data source definition, you must append the security token to the end of the password specified for DataSourcePassword. In the example `secretXaBARTsLZReM4Px47qPLOS`, `secret` is the password and the remainder of the value is the security token.

- All communication between the driver and the Hybrid Data Pipeline service is encrypted using SSL, including the values specified for DataSourceUserID and DataSourcePassword.

**Default**

empty string

**Data Type**

String

**DataSourceUserID**

**Purpose**
Specifies the user ID that is required for logging into a backend data store, such as SQL Server or Salesforce.

**Valid Values**

`user_name`

where:
user_name

is the user ID required for logging into the data store.

Notes

• The data store user ID and password may be stored in the Hybrid Data Pipeline Data Source definition. If that is true and you specify the user ID and password using the DataSourceUserID and DataSourcePassword connection properties, the values specified in these connection properties take precedence.

• All communication between the driver and the Hybrid Data Pipeline service is encrypted using SSL, including the values specified for DataSourceUserID and DataSourcePassword.

Default
empty string

Data Type
String

EnableCancelTimeout

Purpose
Determines whether a cancel request that is sent by the driver as the result of a query timing out is subject to the same query timeout value as the statement it cancels.

Valid Values
ture | false

If set to true, the cancel request times out using the same timeout value, in seconds, that is set for the statement it cancels. For example, if your application calls Statement.setQueryTimeout(5) on a statement and that statement is canceled because its timeout value was exceeded, the driver sends a cancel request that also will time out if its execution exceeds 5 seconds. If the cancel request times out, because the server is down, for example, the driver throws an exception indicating that the cancel request was timed out and the connection is no longer valid.

If set to false, the cancel request does not time out.

Default
false

Data Type
boolean

EncryptionMethod

Purpose
Determines whether data is encrypted and decrypted when transmitted over the network between the driver and database server.
Valid values

noEncryption | SSL

Behavior

If set to noEncryption, data is not encrypted or decrypted.

If set to SSL, data is encrypted using SSL. If the database server does not support SSL, the connection fails and the driver throws an exception.

Notes

• Connection hangs can occur when the driver is configured for SSL and the database server does not support SSL. You may want to set a login timeout using the LoginTimeout property to avoid problems when connecting to a server that does not support SSL.

• When SSL is enabled, the following properties also apply:
  • HostNameInCertificate
  • TrustStore
  • TrustStorePassword
  • ValidateServerCertificate

Default

SSL

Data type

String

HostNameInCertificate

Description

Specifies a host name for certificate validation when SSL encryption is enabled (EncryptionMethod=SSL) and validation is enabled (ValidateServerCertificate=true). This property is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

Valid values

host_name

where:

host_name

is a valid host name.
Behavior

If `host_name` is specified, the driver compares the specified host name to the DNSName value of the SubjectAlternativeName in the certificate. If a DNSName value does not exist in the SubjectAlternativeName or if the certificate does not have a SubjectAlternativeName, the driver compares the host name with the Common Name (CN) part of the certificate’s Subject name. If the values do not match, the connection fails and the driver throws an exception.

Notes

• If SSL encryption or certificate validation is not enabled, this property is ignored.
• If SSL encryption and validation is enabled and this property is unspecified, the driver uses the server name specified in the connection URL or data source of the connection to validate the certificate.

Default

None

Data Type

String

See also

Using data encryption

HybridDataPipelineDataSource

Purpose

REQUIRED. Specifies which Hybrid Data Pipeline Data Source the driver uses for the connection. A Hybrid Data Pipeline Data Source specifies the data store to connect to and the information required to establish a connection to the data store.

The name of the DataSource must be the name of a data source defined in your Hybrid Data Pipeline account. Data Source names are unique within each Hybrid Data Pipeline account; for example, more than one account can have a data source named `test`.

You can create one or more Hybrid Data Pipeline Data Sources in your Hybrid Data Pipeline account. You can create and manage these Data Sources using the Hybrid Data Pipeline Dashboard.

Valid Values

`datasource_name`

where:

`datasource_name`

is the name of a valid Hybrid Data Pipeline Data Source.

Default

None
InsensitiveResultSetBufferSize

Purpose
Determines the amount of memory that is used by the driver to cache insensitive result set data.

Valid Values
-1 | 0 | x

where:

x

is a positive integer that represents the amount of memory.

Behavior
If set to -1, the driver caches insensitive result set data in memory. If the size of the result set exceeds available memory, an OutOfMemoryException is generated. With no need to write result set data to disk, the driver processes the data efficiently.

If set to 0, the driver caches insensitive result set data in memory, up to a maximum of 2 MB. If the size of the result set data exceeds the size of the memory buffer, the driver pages the result set data to disk, which can have a negative performance effect. Because result set data may be written to disk, the driver may have to reformat the data to write it correctly to disk.

If set to x, the driver caches insensitive result set data in memory and uses this value to set the size (in KB) of the memory buffer for caching insensitive result set data. If the size of the result set data exceeds the size of the memory buffer, the driver pages the result set data to disk, which can have a negative performance effect. Because the result set data may be written to disk, the driver may have to reformat the data to write it correctly to disk. Specifying a buffer size that is a power of 2 results in efficient memory use. The maximum cache size setting is 2 GB.

Note: To improve performance when using scroll-insensitive result sets, the driver can caches the result set data in memory instead of writing it to disk. By default, the driver caches 2 MB of insensitive result set data in memory and writes any remaining result set data to disk. Performance can be improved by increasing the amount of memory used by the driver before writing data to disk or by forcing the driver to never write insensitive result set data to disk.

Default
2048

Data Type
int
**JavaDoubleToString**

**Purpose**
Determines which algorithm the driver uses when converting a double or float value to a string value. By default, the driver uses its own internal conversion algorithm, which improves performance.

**Valid Values**
true | false

**Behavior**
If set to true, the driver uses the JVM algorithm when converting a double or float value to a string value. If your application cannot accept rounding differences and you are willing to sacrifice performance, set this value to true to use the JVM conversion algorithm.

If set to false, the driver uses its own internal algorithm when converting a double or float value to a string value. This value improves performance, but slight rounding differences within the allowable error of the double and float data types can occur when compared to the same conversion using the JVM algorithm.

**Default**
false

**Data Type**
boolean

**LogConfigFile**

**Purpose**
Specifies the file name, and optionally, the path of the properties file used to initialize driver logging.

**Valid Values**
string
where:

string

is the relative or fully qualified path of the properties file to load to initialize driver logging. If you do not specify a path, the driver looks for this file in the current working directory. If the specified file does not exist, the driver continues searching for an appropriate properties file as described in Troubleshooting an application by logging on page 749.

**Default**
ddcloudlogging.properties

**Data Type**
String
LoginTimeout

**Purpose**
The amount of time, in seconds, that the driver waits for a connection to be established before timing out the connection request.

**Valid Values**
0 \( x \)

where:

\[ x \]

is a positive integer that represents a number of seconds.

**Behavior**
If set to 0, the driver does not timeout a connection request.
If set to \( x \), the driver waits for the specified number of seconds before returning control to the application and throwing a timeout exception.

**Default**
0

**Data Type**
int

Password

**Description**
Specifies the password to use to connect to the Hybrid Data Pipeline service. A password is required.

---

**Important:** Setting the password using a JDBC data source is not recommended. The JDBC data source persists all properties, including the Password property, in clear text. In contrast, passwords stored in a Hybrid Data Pipeline data source are encrypted.

**Valid Values**

\[ password \]

where:

\[ password \]

is a valid password for the specified Hybrid Data Pipeline service. The password is case-sensitive.
Default
None

Data Type
String

ProxyHost

Description
Identifies a proxy server to use for the connection.

Valid Values
server_name | IP_address
where:

server_name

is the name of the proxy server, which may be qualified with the domain name.

IP_address

is an IP address, specified in either IPv4 or IPv6 format, or a combination of the two.

Notes
• All communication between the driver and the Hybrid Data Pipeline service is encrypted using SSL, including the values specified for DataSourceUserID and DataSourcePassword.

Default
empty string

Data Type
String

See also
Connecting Through a Proxy Server on page 737

ProxyPassword

Purpose
Specifies the password needed to connect to a proxy server. The proxy server is specified by the ProxyHost property.

Valid Values
password
where:

\[ \text{password} \]

is a valid password for that server. Contact your system administrator to obtain a valid password.

**Notes**

- All communication between the driver and the Hybrid Data Pipeline service is encrypted using SSL, including the values specified for DataSourceUserID and DataSourcePassword.

**Default**

empty string

**Data Type**

String

**See also**

- [Connecting Through a Proxy Server](#) on page 737

---

**ProxyPort**

**Purpose**

Specifies the port number where the proxy server is listening for HTTPS requests. The proxy server is specified by the ProxyHost property.

**Valid Values**

\[ \text{port} \]

where:

\[ \text{port} \]

is the port number on which the proxy server is listening. Contact your system administrator to obtain the correct port.

**Default**

empty string

**Data Type**

int

**See also**

- [Connecting Through a Proxy Server](#) on page 737
ProxyUser

**Purpose**
Specifies the user name needed to connect to a proxy server. The proxy server is specified by the ProxyHost property.

**Valid Values**

```
user_name
```

where:

```
user_name
```

is a valid user ID for the proxy server.

**Default**
empty string

**Data Type**
String

**See also**
Connecting Through a Proxy Server on page 737

QueryTimeout

**Purpose**
Sets the default query timeout (in seconds) for all statements created by a connection.

**Valid Values**

```
-1 | 0 | x
```

where:

```
x
```

is a number of seconds.

**Behavior**
If set to -1, the query timeout functionality is disabled. The driver silently ignores calls to the Statement.setQueryTimeout() method.

If set to 0, the default query timeout is infinite (the query does not time out).

If set to x, the driver uses the value as the default timeout for any statement created by the connection. To override the default timeout value that is set by this property, call the Statement.setQueryTimeout() method to set a timeout value for a particular statement.
Default

0

Data Type

int

TransactionMode

Purpose

Specifies how the driver handles manual transactions.

Valid Values

ignore|noTransactions|transactions

Behavior

If set to ignore, the driver always operates in auto-commit mode. Calls to set the driver to manual commit mode and to commit transactions are ignored. Calls to rollback a transaction cause the driver to throw an exception indicating that no transaction is started. Metadata indicates that the driver supports transactions and the ReadUncommitted transaction isolation level.

If set to noTransactions, the driver does not support transactions. Metadata indicates that the driver does not support transactions. Calls to set the driver to manual commit mode, or to commit or rollback transactions, generates an exception.

If set to transactions, the data source and driver support manual transactions for supported data stores. Support for isolation levels depends on which backend data store is being used. If the data store does not support transactions (for example, Salesforce), then TransactionMode is switched to noTransactions.

Default

transactions

Data Type

String

TrustStore

Description

Specifies the directory of the truststore file to be used when SSL is enabled (EncryptionMethod=SSL) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

This value overrides the directory of the truststore file that is specified by the javax.net.ssl.trustStore Java system property. If this property is not specified, the truststore directory is specified by the javax.net.ssl.trustStore Java system property.

This property is ignored if ValidateServerCertificate=false.
Valid values

*string*

*string*

is the directory of the truststore file.

**Default**

None

**Data Type**

String

**See also**

Using data encryption

## TrustStorePassword

**Description**

Specifies the password that is used to access the truststore file when SSL is enabled (EncryptionMethod=SSL) and server authentication is used. The truststore file contains a list of the Certificate Authorities (CAs) that the client trusts.

This property is ignored if ValidateServerCertificate=false.

**Valid values**

*string*

where:

*string*

is a valid password for the truststore file.

**Notes**

- This value overrides the password of the truststore file that is specified by the javax.net.ssl.trustStorePassword Java system property. If this property is not specified, the truststore password is specified by the javax.net.ssl.trustStorePassword Java system property.

- This property is ignored if ValidateServerCertificate=false.

**Default**

None

**Data type**

String

**See also**

Using data encryption
**User**

**Purpose**
Specifies the user name that is used to connect to the Hybrid Data Pipeline service. A user name is required.

**Valid Values**

```
string
```

where:

```
string
```

is a valid user name for the specified Hybrid Data Pipeline service. The user name is case-insensitive.

**Default**
None

**Data Type**
String

**ValidateServerCertificate**

**Description**
Determines whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled (EncryptionMethod=SSL). When using SSL server authentication, any certificate that is sent by the server must be issued by a trusted Certificate Authority (CA).

**Valid values**

```
true | false
```

**Behavior**
If set to `true`, the driver validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted CA in the truststore file. If the HostNameInCertificate property is specified, the driver also validates the certificate using a host name. The HostNameInCertificate property is optional and provides additional security against man-in-the-middle (MITM) attacks by ensuring that the server the driver is connecting to is the server that was requested.

If set to `false`, the driver does not validate the certificate that is sent by the database server. The driver ignores any truststore information that is specified by the TrustStore and TrustStorePassword properties or Java system properties.

**Notes**

- Truststore information is specified using the TrustStore and TrustStorePassword properties or by using Java system properties.
• Allowing the driver to trust any certificate that is returned from the server even if the issuer is not a trusted CA is useful in test environments because it eliminates the need to specify truststore information on each client in the test environment.

Default
true

Data type
String

See also
Using data encryption

WSRetryCount

Purpose
The number of times the driver retries a timed-out Select request. Insert, Update, and Delete requests are never retried. The timeout period is specified by the WSTimeout connection property.

Valid Values
0 | x
where:

x
is a positive integer.

Behavior
If set to 0, the driver does not retry timed-out requests after the initial unsuccessful attempt.
If set to x, the driver retries the timed-out requests the specified number of times.

Example
If this property is set to 2, the driver retries the connection attempt twice after the initial attempt.

Default
0

Data Type
int
WSRetryDelay

**Purpose**
Specifies the time, in seconds, that the driver waits for a response to a Web service request.

**Valid Values**
0 | x
where:

x

is a number of seconds.

**Behavior**
If set to 0, the driver does not delay between retries.
If set to x, the driver waits between connection retry attempts the specified number of seconds.

**Example**
If WSRetryCount is set to 2 and this property is set to 3, the driver retries the Hybrid Data Pipeline Data Source twice after the initial retry attempt. The driver waits 3 seconds between retry attempts.

**Default**
1 (second)

**Data Type**
int

**JDBC support**
The Hybrid Data Pipeline Driver for JDBC is compatible with JDBC 2.0, 3.0, 4.0, 4.1, and 4.2. The following topics describe support for JDBC interfaces and methods.

**Note:** In this section, the phrase "Salesforce-type data stores" includes Salesforce, Force.com, Database.com, ServiceMax, Veeva CRM, and FinancialForce.

**Array**

<table>
<thead>
<tr>
<th>Array Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void free()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### Array Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object getArray()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Object getArray(Map)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
<tr>
<td>Object getArray(long, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Object getArray(long, int, Map)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
<tr>
<td>int getBaseType()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getBaseTypeName()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getResultSet()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getResultSet(Map)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
<tr>
<td>ResultSet getResultSet(long, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getResultSet(long, int, Map)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
</tbody>
</table>

### Blob

<table>
<thead>
<tr>
<th>Method</th>
<th>Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void free()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>InputStream getBinaryStream()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>byte[] getBytes(long, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>long length()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>long position(Blob, long)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>Blob Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>long position(byte[], long)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LONGVARBINARY data type.</td>
</tr>
<tr>
<td>OutputStream setBinaryStream(long)</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LONGVARBINARY data type.</td>
</tr>
<tr>
<td>int setBytes(long, byte[])</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LONGVARBINARY data type.</td>
</tr>
<tr>
<td>int setBytes(long, byte[], int, int)</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LONGVARBINARY data type.</td>
</tr>
<tr>
<td>void truncate(long)</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LONGVARBINARY data type.</td>
</tr>
</tbody>
</table>

**CallableStatement**

<table>
<thead>
<tr>
<th>CallableStatement Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array getArray(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>Array getArray(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Reader getCharacterStream(int)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>Reader getCharacterStream(String)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>BigDecimal getBigDecimal(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>CallableStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>BigDecimal getBigDecimal(int, int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce.com, Force.com, Database.com: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>BigDecimal getBigDecimal(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Blob getBlob(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type. Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>Blob getBlob(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>boolean getBoolean(int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>boolean getBoolean(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>byte getByte(int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception when an application calls output parameters.</td>
</tr>
<tr>
<td>byte getByte(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>byte [] getBytes(int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>byte [] getBytes(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>CallableStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Clob getClob(int)              | 2.0 Core          | Yes       | The driver supports using with data types that map to the JDBC `LONGVARCHAR` data type.  
Salesforce-type data stores: The driver throws an "invalid parameter bindings" exception if your application calls output parameters. |
<p>| Clob getClob(String)           | 3.0               | No        | The driver throws an &quot;unsupported method&quot; exception.                                                                                     |
| Date getDate(int)              | 1.0               | Yes       | Salesforce: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.                     |
| Date getDate(int, Calendar)    | 2.0 Core          | Yes       | Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.       |
| Date getDate(String)            | 3.0               | No        | The driver throws an &quot;unsupported method&quot; exception.                                                                                     |
| Date getDate(String, Calendar) | 3.0               | No        | The driver throws an &quot;unsupported method&quot; exception.                                                                                     |
| double getDouble(int)          | 1.0               | Yes       | Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.       |
| double getDouble(String)       | 3.0               | No        | The driver throws an &quot;unsupported method&quot; exception.                                                                                     |
| float getFloat(int)            | 1.0               | Yes       | Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.       |
| float getFloat(String)         | 3.0               | No        | The driver throws an &quot;unsupported method&quot; exception.                                                                                     |</p>
<table>
<thead>
<tr>
<th>CallableStatement Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int getInt(int)</code></td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td><code>int getInt(String)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>long getLong(int)</code></td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td><code>long getLong(String)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>Reader getNCharacterStream(int)</code></td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>Reader getNCharacterStream(String)</code></td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>NClob getNClob(int)</code></td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>NClob getNClob(String)</code></td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>String getNString(int)</code></td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>String getNString(String)</code></td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>Object getObject(int)</code></td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td><code>Object getObject(int, Map)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
<tr>
<td><code>Object getObject(String)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>Object getObject(String, Map)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>CallableStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>Ref getRef(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>Ref getRef(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>short getShort(int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>short getShort(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>SQLXML getSQLXML(int)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>SQLXML getSQLXML(String)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>String getString(int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>String getString(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Time getTime(int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>Time getTime(int, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>Time getTime(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Time getTime(String, Calendar)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>CallableStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>Timestamp getTimestamp(int)</code></td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td><code>Timestamp getTimestamp(int, Calendar)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td><code>Timestamp getTimestamp(String)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>Timestamp getTimestamp(String, Calendar)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>URL getURL(int)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>URL getURL(String)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>boolean isWrapperFor(Class&lt;?&gt; iface)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void registerOutParameter(int, int)</code></td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td><code>void registerOutParameter(int, int, int)</code></td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td><code>void registerOutParameter(int, int, String)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the <code>String</code> parameter. Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>CallableStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>void registerOutParameter(String, int)</td>
<td>3.0</td>
<td>Yes</td>
<td>Salesforce.com, Force.com, Database.com: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>void registerOutParameter(String, int, int)</td>
<td>3.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>void registerOutParameter(String, int, String)</td>
<td>3.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws an &quot;invalid parameter bindings&quot; exception if your application calls output parameters.</td>
</tr>
<tr>
<td>void setArray(int, Array)</td>
<td>2.0 Core</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setAsciiStream(String, InputStream)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setAsciiStream(String, InputStream, int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setAsciiStream(String, InputStream, long)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBigDecimal(String, BigDecimal)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBinaryStream(String, InputStream)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBinaryStream(String, InputStream, int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBinaryStream(String, InputStream, long)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBlob(String, Blob)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBlob(String, InputStream)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBlob(String, InputStream, long)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>CallableStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>void setBoolean(String, boolean)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setByte(String, byte)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setBytes(String, byte [])</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setCharacterStream(String, Reader, int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setCharacterStream(String, InputStream, long)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setClob(String, Clob)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setClob(String, Reader, Clob)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setClob(String, Reader, long)</td>
<td>4.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setDate(String, Date)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setDate(String, Date, Calendar)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setDouble(String, double)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setFloat(String, float)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setInt(String, int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setLong(String, long)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setNCharacterStream(String, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setNClob(String, NClob)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setNClob(String, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setNClob(String, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>CallableStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td><code>void setNString(String, String)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void setNull(int, int, String)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void setNull(String, int)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setNull(String, int, String)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setObject(String, Object)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setObject(String, Object, int)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setObject(String, Object, int, int)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setShort(String, short)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setSQLXML(String, SQLXML)</code></td>
<td>4.0</td>
<td>No</td>
<td>The driver throws &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setString(String, String)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setTime(String, Time)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setTime(String, Time, Calendar)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setTimestamp(String, Timestamp)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setTimestamp(String, Timestamp, Calendar)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void setURL(String, URL)</code></td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>boolean wasNull()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
# Clob

<table>
<thead>
<tr>
<th>Clob Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void free()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>InputStream getAsciiStream()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>Reader getCharacterStream()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>Reader getCharacterStream(long, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>String getSubString(long, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>long length()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>long position(Clob, long)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>long position(String, long)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>OutputStream setAsciiStream(long)</td>
<td>3.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>Writer setCharacterStream(long)</td>
<td>3.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>int setString(long, String)</td>
<td>3.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
</tbody>
</table>
### Clob Methods

<table>
<thead>
<tr>
<th>Method Description</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>int setString(long, String, int, int)</td>
<td>3.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC <code>LONGVARCHAR</code> data type.</td>
</tr>
<tr>
<td>void truncate(long)</td>
<td>3.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC <code>LONGVARCHAR</code> data type.</td>
</tr>
</tbody>
</table>

### Connection

#### Connection Methods

<table>
<thead>
<tr>
<th>Method Description</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void clearWarnings()</td>
<td>1.0</td>
<td>Yes</td>
<td>If a connection is closed while a transaction is still active, that transaction is rolled back.</td>
</tr>
<tr>
<td>void close()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void commit()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Blob createBlob()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Clob createClob()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>NClob createNClob()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>SQLXML createSQLXML()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Statement createStatement()</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: Scroll-sensitive result sets are expensive from both a Web service call perspective and a performance perspective. The driver expends a network round trip for each row that is fetched.</td>
</tr>
<tr>
<td>Statement createStatement(int, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Statement createStatement(int, int, int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Struct createStruct(String, Object[])</td>
<td>1.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>boolean getAutoCommit()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Connection Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>String getCatalog()</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver returns an empty string because this data store does not have the concept of a catalog.</td>
</tr>
<tr>
<td>String getClientInfo()</td>
<td>4.0</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>String getClientInfo(String)</td>
<td>4.0</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>int getHoldability()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DatabaseMetaData metaData()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getTransactionIsolation()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Map getTypeMap()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver always returns an empty java.util.HashMap.</td>
</tr>
<tr>
<td>SQLWarning getWarnings()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isClosed()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isReadOnly()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isValid()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWrapperFor(Class&lt;?&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String nativeSQL(String)</td>
<td>1.0</td>
<td>Yes</td>
<td>The driver always returns the same value that was passed in from the application.</td>
</tr>
<tr>
<td>CallableStatement prepareCall(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>CallableStatement prepareCall(String, int, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver downgrades ResultSet.TYPE_SCROLL_SENSITIVE to TYPE_SCROLL_INSENSITIVE.</td>
</tr>
<tr>
<td>CallableStatement prepareCall(String, int, int, int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>PreparedStatement prepareStatement (String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>PreparedStatement prepareStatement (String, int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Connection Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>PreparedStatement prepareStatement (String, int, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: Scroll-sensitive result sets are expensive from both a Web service call perspective and a performance perspective. The driver expends a network round trip for each row that is fetched.</td>
</tr>
<tr>
<td>PreparedStatement prepareStatement (String, int, int, int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>PreparedStatement prepareStatement (String, int[])</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>PreparedStatement prepareStatement (String, String [])</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void releaseSavepoint(Savepoint)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void rollback()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void rollback(Savepoint)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setAutoCommit(boolean)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver throws a &quot;transactions not supported&quot; exception if set to false.</td>
</tr>
<tr>
<td>void setCatalog(String)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver ignores any value set by the String argument because this data store does not have the concept of a catalog.</td>
</tr>
<tr>
<td>String setClientInfo(Properties)</td>
<td>4.0</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>String setClientInfo(String, String)</td>
<td>4.0</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>void setHoldability(int)</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver ignores this method.</td>
</tr>
<tr>
<td>void setReadOnly(boolean)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Savepoint setSavepoint()</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Savepoint setSavepoint(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
</tbody>
</table>
### Connection Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void setTransactionIsolation(int)</td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver ignores the specified transaction isolation level.</td>
</tr>
<tr>
<td>void setTypeMap(Map)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
<tr>
<td>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### ConnectionEventListener

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void connectionClosed(event)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void connectionErrorOccurred(event)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### ConnectionPoolDataSource

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>int getLoginTimeout()</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>PrintWriter getLogWriter()</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>PooledConnection getPooledConnection()</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>PooledConnection getPooledConnection(String, String)</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setLoginTimeout(int)</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setLogWriter(PrintWriter)</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
## DatabaseMetaData

<table>
<thead>
<tr>
<th>DatabaseMetaData Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean autoCommitFailureClosesAllResultSets()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean allProceduresAreCallable()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean allTablesAreSelectable()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean dataDefinitionCausesTransactionCommit()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean dataDefinitionIgnoredInTransactions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean deletesAreDetected(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean doesMaxRowSizeIncludeBlobs()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>getAttributes(String, String, String, String)</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver returns an empty result set.</td>
</tr>
<tr>
<td>ResultSet getBestRowIdentifier(String, String, String, int, boolean)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getCatalogs()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getCatalogSeparator()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getCatalogTerm()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getClientInfoProperties()</td>
<td>4.0</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>ResultSet getColumnPrivileges(String, String, String, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getColumns(String, String, String, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Connection getConnection()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getCrossReference(String, String, String, String, String, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getFunctions()</td>
<td>4.0</td>
<td>Yes</td>
<td>The driver returns an empty result set.</td>
</tr>
<tr>
<td>DatabaseMetaData Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>ResultSet getFunctionColumns()</td>
<td>4.0</td>
<td>Yes</td>
<td>The driver returns an empty result set.</td>
</tr>
<tr>
<td>int getDatabaseMajorVersion()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getDatabaseMinorVersion()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getDatabaseProductName()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getDatabaseProductVersion()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getDefaultTransactionIsolation()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getDriverMajorVersion()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getDriverMinorVersion()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getDriverName()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getDriverVersion()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getExportedKeys(String, String, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getExtraNameCharacters()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getIdentifierQuoteString()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getImportedKeys(String, String, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getIndexInfo(String, String, String, boolean, boolean)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getJDBCMajorVersion()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getJDBCMinorVersion()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMaxBinaryLiteralLength()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMaxCatalogNameLength()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMaxCharLiteralLength()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMaxColumnNameLength()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMaxColumnsInGroupBy()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMaxColumnsInIndex()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DatabaseMetaData Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>int getMaxColumnsInOrderBy()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxColumnsInSelect()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxColumnsInTable()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxConnections()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxCursorNameLength()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxIndexLength()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxProcedureNameLength()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxRowSize()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxSchemaNameLength()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxStatementLength()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxStatements()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxTableNameLength()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxTablesInSelect()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getMaxUserNameLength()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>String getNumericFunctions()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>ResultSet getPrimaryKeys(String, String, String)</code></td>
<td>1.0</td>
<td>Yes</td>
<td>Salesforce: SchemaName and ProcedureName must be explicit values; they cannot be patterns.</td>
</tr>
<tr>
<td><code>ResultSet getProcedureColumns(String, String, String, String)</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>ResultSet getProcedures(String, String, String)</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>String getProcedureTerm()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int getResultSetHoldability()</code></td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>ResultSet getSchemas()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>ResultSet getSchemas(catalog, pattern)</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DatabaseMetaData Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>String getSchemaTerm()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getSearchStringEscape()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getSQLKeywords()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getSQLStateType()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getStringFunctions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getSuperTables(String, String, String)</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver returns an empty result set.</td>
</tr>
<tr>
<td>ResultSet getSuperTypes(String, String, String)</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver returns an empty result set.</td>
</tr>
<tr>
<td>String getSystemFunctions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getTablePrivileges(String, String, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getTables(String, String, String, String [])</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getTableTypes()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getTimeDateFunctions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getTypeInfo()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getUDTs(String, String, String, int [])</td>
<td>2.0 Core</td>
<td>No</td>
<td>The driver returns an empty result set.</td>
</tr>
<tr>
<td>String getURL()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getUserName()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getVersionColumns(String, String, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean insertsAreDetected(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isCatalogAtStart()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isReadOnly()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWrapperFor(Class&lt;? extends iFace)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean locatorsUpdateCopy()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DatabaseMetaData Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>boolean nullsAreSortedAtEnd()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean nullsAreSortedAtStart()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean nullsAreSortedHigh()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean nullsAreSortedLow()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean othersDeletesAreVisible(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean othersInsertsAreVisible(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean othersUpdatesAreVisible(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean ownDeletesAreVisible(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean ownInsertsAreVisible(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean ownUpdatesAreVisible(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean storesLowerCaseIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean storesLowerCaseQuotedIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean storesUpperCaseIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean storesUpperCaseQuotedIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean storesMixedCaseIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean storesMixedCaseQuotedIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsAlterTableWithAddColumn()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsAlterTableWithDropColumn()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsANSI92EntryLevelSQL()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsANSI92FullSQL()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsANSI92IntermediateSQL()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsBatchUpdates()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsCatalogsInDataManipulation()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DatabaseMetaData Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>boolean supportsCatalogsInIndexDefinitions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsCatalogsInPrivilegeDefinitions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsCatalogsInProcedureCalls()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsCatalogsInTableDefinitions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsColumnAliasing()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsConvert()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsConvert(int, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsCoreSQLGrammar()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsCorrelatedSubqueries()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsDataDefinitionAndDataManipulationTransactions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsDataManipulationTransactionsOnly()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsDifferentTableCorrelationNames()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsExpressionsInOrderBy()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsExtendedSQLGrammar()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsFullOuterJoins()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsGetGeneratedKeys()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsGroupBy()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsGroupByBeyondSelect()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsGroupByUnrelated()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsIntegrityEnhancementFacility()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsLikeEscapeClause()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DatabaseMetaData Methods</td>
<td>Version</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>boolean supportsLimitedOuterJoins()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsMinimumSQLGrammar()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsMixedCaseIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsMixedCaseQuotedIdentifiers()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsMultipleOpenResults()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsMultipleResultSets()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsMultipleTransactions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsNamedParameters()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsNonNullableColumns()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsOpenCursorsAcrossCommit()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsOpenCursorsAcrossRollback()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsOpenStatementsAcrossCommit()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsOpenStatementsAcrossRollback()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsOrderByUnrelated()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsOuterJoins()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsPositionedDelete()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsPositionedUpdate()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsResultSetConcurrency(int, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsResultSetHoldability(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsResultSetType(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSavePoints()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSchemasInDataManipulation()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DatabaseMetaData Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>boolean supportsSchemasInIndexDefinitions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSchemasInPrivilegeDefinitions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSchemasInProcedureCalls()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSchemasInTableDefinitions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSelectForUpdate()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsStoredFunctionsUsingCallSyntax()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsStoredProcedures()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSubqueriesInComparisons()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSubqueriesInExists()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSubqueriesInIns()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsSubqueriesInQuantifieds()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsTableCorrelationNames()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsTransactionIsolationLevel(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsTransactions()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsUnion()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean supportsUnionAll()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>(&lt;T&gt; T) unwrap(Class(&lt;T&gt;) iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean updatesAreDetected(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean usesLocalFilePerTable()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean usesLocalFiles()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### DataSource

<table>
<thead>
<tr>
<th>DataSource Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection getConnection()</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Connection getConnection(String, String)</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getLoginTimeout()</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>PrintWriter getLogWriter()</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWrapperFor(Class&lt;?&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setLoginTimeout(int)</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setLogWriter(PrintWriter)</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The DataSource interface implements the javax.naming.Referenceable and java.io.Serializable interfaces.

### Driver

<table>
<thead>
<tr>
<th>Driver Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean acceptsURL(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Connection connect(String, Properties)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMajorVersion()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMinorVersion()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DriverPropertyInfo [] getParameterInfo(String, Properties)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### ParameterMetaData

<table>
<thead>
<tr>
<th>ParameterMetaData Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getParameterClassName(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getParameterCount()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getParameterMode(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getParameterType(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getParameterTypeName(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getPrecision(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getScale(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int isNullable(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isSigned(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWrapperFor(Class&lt;?&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean jdbcCompliant()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### PooledConnection

<table>
<thead>
<tr>
<th>PooledConnection Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void addConnectionEventListener(listener)</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void addStatementEventListener(listener)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void close()</td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### PooledConnection Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Connection getConnection()</code></td>
<td>2.0 Optional</td>
<td>Yes</td>
<td>A pooled connection object can have only one Connection object open (the one most recently created). Depending on your PoolManager implementation, the application can invoke this method a second time as a way to take a connection away from an application and give it to another user (a rare occurrence). The driver does not support the &quot;reclaiming&quot; of connections and will throw an exception.</td>
</tr>
<tr>
<td><code>void removeConnectionEventListener(listener)</code></td>
<td>2.0 Optional</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void removeStatementEventListener(listener)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### PreparedStatement

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void addBatch()</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void clearParameters()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>boolean execute()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>ResultSet executeQuery()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>int executeUpdate()</code></td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>ResultSetMetaData getMetaData()</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>ParameterMetaDatagetParameterMetaData()</code></td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>boolean isWrapperFor(Class&lt;?&gt; iface)</code></td>
<td>4.0</td>
<td>Yes</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td><code>void setArray(int, Array)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void setAsciiStream(int, InputStream)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>PreparedStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>----------------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void setAsciiStream(int, InputStream, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setAsciiStream(int, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setBigDecimal(int, BigDecimal)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setBinaryStream(int, InputStream)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setBinaryStream(int, InputStream, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setBinaryStream(int, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setBlob(int, Blob)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>void setBlob(int, InputStream)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>void setBlob(int, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>void setBoolean(int, boolean)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setByte(int, byte)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setBytes(int, byte [])</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setCharacterStream(int, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setCharacterStream(int, Reader, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setCharacterStream(int, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setClob(int, Clob)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>PreparedStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void setClob(int, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>void setClob(int, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>void setDate(int, Date)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setDate(int, Date, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setDouble(int, double)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setFloat(int, float)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setInt(int, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setLong(int, long)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setNCharacterStream(int, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void setNCharacterStream(int, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void setNClob(int, NClob)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void setNClob(int, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void setNClob(int, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void setNull(int, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setNull(int, int, String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setNString(int, String)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setObject(int, Object)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>PreparedStatement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>----------------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>void setObject(int, Object, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setObject(int, Object, int, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setQueryTimeout(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setRef(int, Ref)</td>
<td>2.0 Core</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setShort(int, short)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setSQLXML(int, SQLXML)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setString(int, String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setTime(int, Time)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setTime(int, Time, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setTimestamp(int, Timestamp)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setTimestamp(int, Timestamp, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setUnicodeStream(int, InputStream, int)</td>
<td>1.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception. This method was deprecated in JDBC 2.0.</td>
</tr>
<tr>
<td>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setURL(int, URL)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
</tbody>
</table>

**Ref**

<table>
<thead>
<tr>
<th>Ref Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>(all)</td>
<td>2.0 Core</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
### ResultSet

<table>
<thead>
<tr>
<th>ResultSet Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean absolute(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void afterLast()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void beforeFirst()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void cancelRowUpdates()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void clearWarnings()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void close()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void deleteRow()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int findColumn(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean first()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Array getArray(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Array getArray(String)</td>
<td>2.0 Core</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>InputStream getAsciiStream(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>InputStream getAsciiStream(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>BigDecimal getBigDecimal(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>BigDecimal getBigDecimal(int, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>BigDecimal getBigDecimal(String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>BigDecimal getBigDecimal(String, int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>InputStream getBinaryStream(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>InputStream getBinaryStream(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Blob getBlob(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>Blob getBlob(String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>ResultSet Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>boolean getBoolean(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean getBoolean(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>byte getByte(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>byte getByte(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>byte [] getBytes(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>byte [] getBytes(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Reader getCharacterStream(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Reader getCharacterStream(String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Clob getClob(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>Clob getClob(String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARBINARY data type.</td>
</tr>
<tr>
<td>int getConcurrency()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getCursorName()</td>
<td>1.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Date getDate(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Date getDate(int, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Date getDate(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Date getDate(String, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>double getDouble(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>double getDouble(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getFetchDirection()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getFetchSize()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>float getFloat(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>float getFloat(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>int getHoldability()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getInt(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getInt(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>long getLong(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>long getLong(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSetMetaData getMetaData()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Reader getNCharacterStream(int)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Reader getNCharacterStream(String)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>NClob getNClob(int)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>NClob getNClob(String)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getNString(int)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getNString(String)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Object getObject(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Object getObject(int, Map)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
<tr>
<td>Object getObject(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Object getObject(String, Map)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver ignores the Map parameter.</td>
</tr>
<tr>
<td>Ref getRef(int)</td>
<td>2.0 Core</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Ref getRef(String)</td>
<td>2.0 Core</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>intgetRow()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>short getShort(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>short getShort(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>SQLXML getSQLXML(int)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>SQLXML getSQLXML(String)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>----------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Statement getStatement()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getString(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getString(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Time getTime(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Time getTime(int, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Time getTime(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Time getTime(String, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Timestamp getTimestamp(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Timestamp getTimestamp(int, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Timestamp getTimestamp(String)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Timestamp getTimestamp(String, Calendar)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getType()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>InputStream getUnicodeStream(int)</td>
<td>1.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception. This method was deprecated in JDBC 2.0.</td>
</tr>
<tr>
<td>InputStream getUnicodeStream(String)</td>
<td>1.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception. This method was deprecated in JDBC 2.0.</td>
</tr>
<tr>
<td>URL getURL(int)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>URL getURL(String)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>SQLWarning getWarnings()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void insertRow()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isAfterLast()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isBeforeFirst()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isClosed()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isFirst()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>boolean isLast()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWrapperFor(Class&lt;? &gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean last()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void moveToCurrentRow()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void moveToInsertRow()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean next()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean previous()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void refreshRow()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean relative(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean rowDeleted()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean rowInserted()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean rowUpdated()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setFetchDirection(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setFetchSize(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateArray(int, Array)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void updateArray(String, Array)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void updateAsciiStream(int, InputStream, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateAsciiStream(int, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateAsciiStream(String, InputStream)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateAsciiStream(String, InputStream, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>void updateAsciiStream(String, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBigDecimal(int, BigDecimal)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBigDecimal(String, BigDecimal)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBinaryStream(int, InputStream)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBinaryStream(int, InputStream, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBinaryStream(int, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBinaryStream(String, InputStream)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBinaryStream(String, InputStream, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBinaryStream(String, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBlob(int, Blob)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBlob(int, InputStream)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBlob(int, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBlob(String, Blob)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBlob(String, InputStream)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateBlob(String, InputStream, long)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### ResultSet Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void updateBoolean(int, boolean)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateBoolean(String, boolean)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateByte(int, byte)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateByte(String, byte)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateBytes(int, byte [])</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateBytes(String, byte [])</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateCharacterStream(int, Reader)</code></td>
<td>4.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td><code>void updateCharacterStream(int, Reader, int)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateCharacterStream(int, Reader, long)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateCharacterStream(String, Reader)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateCharacterStream(String, Reader, int)</code></td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateCharacterStream(String, Reader, long)</code></td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><code>void updateClob(int, Clob)</code></td>
<td>3.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td><code>void updateClob(int, Reader)</code></td>
<td>4.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td><code>void updateClob(int, Reader, long)</code></td>
<td>4.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td><code>void updateClob(String, Clob)</code></td>
<td>3.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td><code>void updateClob(String, Reader)</code></td>
<td>4.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>ResultSet Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void updateClob(String, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>The driver supports using with data types that map to the JDBC LONGVARCHAR data type.</td>
</tr>
<tr>
<td>void updateDate(int, Date)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateDate(String, Date)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateDouble(int, double)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateDouble(String, double)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateFloat(int, float)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateFloat(String, float)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateInt(int, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateInt(String, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateLong(int, long)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateLong(String, long)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateNClob(int, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(int, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(String, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(String, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(int, NClob)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(int, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>ResultSet Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void updateNClob(int, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(String, NClob)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(String, Reader)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNClob(String, Reader, long)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNString(int, String)</td>
<td>4.0</td>
<td>Yes</td>
<td>Salesforce-type data sources: N methods are identical to their non-N counterparts.</td>
</tr>
<tr>
<td>void updateNString(String, String)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>void updateNull(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateNull(String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateObject(int, Object)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateObject(int, Object, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateObject(String, Object)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateObject(String, Object, int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateRef(int, Ref)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void updateRef(String, Ref)</td>
<td>3.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void updateRow()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateShort(int, short)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateShort(String, short)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateSQLXML(int, SQLXML)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### ResultSet Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void updateSQLXML(String, SQLXML)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateString(int, String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateString(String, String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateTime(int, Time)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateTime(String, Time)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void updateTime(Timestamp, Timestamp)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean wasNull()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### ResultSetMetaData

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getCatalogName(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getColumnClassName(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getColumnCount()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getColumnDisplaySize(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getColumnLabel(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getColumnName(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getColumnType(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getColumnTypeName(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getPrecision(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getScale(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getSchemaName(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>String getTableName(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isAutoIncrement(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### ResultSetMetaData Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean isCaseSensitive(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isCurrency(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isDefinitelyWritable(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int isNullable(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isReadOnly(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isSearchable(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isSigned(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWrapperFor(Class&lt;?&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWritable(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### RowSet

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>(all)</td>
<td>2.0 Optional</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

### SavePoint

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>(all)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### Statement

<table>
<thead>
<tr>
<th>Method</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void addBatch(String)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td>The driver throws an &quot;invalid method call&quot; exception for PreparedStatement and CallableStatement.</td>
</tr>
<tr>
<td>Statement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>void cancel()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void clearBatch()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void clearWarnings()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void close()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean execute(String)</td>
<td>1.0</td>
<td>Yes</td>
<td>The driver throws an &quot;invalid method call&quot; exception for PreparedStatement and CallableStatement.</td>
</tr>
<tr>
<td>boolean execute(String, int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean execute(String, int [])</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>boolean execute(String, String [])</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>int [] executeBatch()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet executeQuery(String)</td>
<td>1.0</td>
<td>Yes</td>
<td>The driver throws an &quot;invalid method call&quot; exception for PreparedStatement and CallableStatement.</td>
</tr>
<tr>
<td>int executeUpdate(String)</td>
<td>1.0</td>
<td>Yes</td>
<td>The driver throws an &quot;invalid method call&quot; exception for PreparedStatement and CallableStatement.</td>
</tr>
<tr>
<td>int executeUpdate(String, int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int executeUpdate(String, int [])</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>int executeUpdate(String, String [])</td>
<td>3.0</td>
<td>Yes</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>Connection getConnection()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getFetchDirection()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getFetchSize()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>ResultSet getGeneratedKeys()</td>
<td>3.0</td>
<td>Yes</td>
<td>Salesforce-type data stores: The driver returns the ID of the last row that was inserted.</td>
</tr>
<tr>
<td>Statement Methods</td>
<td>Version Introduced</td>
<td>Supported</td>
<td>Comments</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>int getMaxFieldSize()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getMaxRows()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean getMoreResults()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean getMoreResults(int)</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getQueryTimeout()</td>
<td>1.0</td>
<td>Yes</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>ResultSet getResultSet()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getResultSetConcurrency()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getResultSetHoldability()</td>
<td>3.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getResultsetType()</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>int getUpdateCount()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>SQLWarning getWarnings()</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isClosed()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isPoolable()</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>boolean isWrapperFor(Class&lt;?&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setCursorName(String)</td>
<td>1.0</td>
<td>No</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
<tr>
<td>void setEscapeProcessing(boolean)</td>
<td>1.0</td>
<td>Yes</td>
<td>The driver ignores this method.</td>
</tr>
<tr>
<td>void setFetchDirection(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setFetchSize(int)</td>
<td>2.0 Core</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setMaxFieldSize(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setMaxRows(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setPoolable(boolean)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void setQueryTimeout(int)</td>
<td>1.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>&lt;T&gt; T unwrap(Class&lt;T&gt; iface)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
### StatementEventListener

<table>
<thead>
<tr>
<th>StatementEventListener Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>void statementClosed(event)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>void statementErrorOccurred(event)</td>
<td>4.0</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### Struct

<table>
<thead>
<tr>
<th>Struct Methods</th>
<th>Version Introduced</th>
<th>Supported</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>(all)</td>
<td>2.0</td>
<td>Yes</td>
<td>The driver throws an &quot;unsupported method&quot; exception.</td>
</tr>
</tbody>
</table>

### DataDirect connection pooling

Hybrid Data Pipeline Driver for JDBC provides a connection pool implementation. This section describes the interfaces and connection methods.

### DataDirect Connection Pool Manager interfaces

This section describes DataDirect Connection Pool Manager interfaces and their methods.

### PooledConnectionDataSource

The PooledConnectionDataSource interface is used to create a PooledConnectionDataSource object for use with the DataDirect Connection Pool Manager.

<table>
<thead>
<tr>
<th>PooledConnectionDataSource Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void close()</td>
<td>Closes the connection pool. All physical connections in the pool are closed. Any subsequent connection request re-initializes the connection pool.</td>
</tr>
<tr>
<td>Connection getConnection()</td>
<td>Obtains a physical connection from the connection pool.</td>
</tr>
<tr>
<td>Connection getConnection(String user, String password)</td>
<td>Obtains a physical connection from the connection pool, where user is the user requesting the connection and password is the password for the connection.</td>
</tr>
<tr>
<td>String getDataSourceName()</td>
<td>Returns the JNDI name that is used to look up the DataDirect DataSource object referenced by this PooledConnectionDataSource.</td>
</tr>
<tr>
<td>PooledConnectionDataSource Methods</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>String getDescription()</td>
<td>Returns the description of this PooledConnectionDataSource.</td>
</tr>
<tr>
<td>int getInitialPoolSize()</td>
<td>Returns the value of the initial pool size, which is the number of physical connections created when the connection pool is initialized.</td>
</tr>
<tr>
<td>int getLoginTimeout()</td>
<td>Returns the value of the login timeout, which is the time allowed for the data store login to be validated.</td>
</tr>
<tr>
<td>PrintWriter getLogWriter()</td>
<td>Returns the writer to which the Pool Manager sends trace information about its activities.</td>
</tr>
<tr>
<td>int getMaxIdleTime()</td>
<td>Returns the value of the maximum idle time, which is the time a physical connection can remain idle in the connection pool before it is removed from the connection pool.</td>
</tr>
<tr>
<td>int getMaxPoolSize()</td>
<td>Returns the value of the maximum pool size. See Configuring pool size on page 820 for more information about how the Pool Manager implements the maximum pool size.</td>
</tr>
<tr>
<td>int getMaxPoolSizeBehavior()</td>
<td>Returns the value of the maximum pool size behavior. See Configuring pool size on page 820 for more information about how the Pool Manager implements the maximum pool size.</td>
</tr>
<tr>
<td>int getMinPoolSize()</td>
<td>Returns the value of the minimum pool size, which is the minimum number of idle connections to be kept in the pool.</td>
</tr>
<tr>
<td>int getPropertyCycle()</td>
<td>Returns the value of the property cycle, which specifies how often the pool maintenance thread wakes up and checks the connection pool.</td>
</tr>
<tr>
<td>Reference getReference()</td>
<td>Obtains a javax.naming.Reference object for this PooledConnectionDataSource. The Reference object contains all the state information needed to recreate an instance of this data source using the PooledConnectionDataSourceFactory object. This method is typically called by a JNDI service provider when this PooledConnectionDataSource is bound to a JNDI naming service.</td>
</tr>
<tr>
<td>public static ConnectionPoolMonitor[] getMonitor()</td>
<td>Returns an array of Connection Pool Monitors, one for each connection pool managed by the Pool Manager.</td>
</tr>
</tbody>
</table>
### PooledConnectionDataSource Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public static ConnectionPoolMonitor getMonitor(String name)</td>
<td>Returns the name of the Connection Pool Monitor for the connection pool specified by <code>name</code>. If a pool with the specified name cannot be found, this method returns null. The connection pool name has the form: <code>jndi_name-user_id</code> where <code>jndi_name</code> is the name used for the JNDI lookup of the driver <code>DataSource</code> object from which the pooled connection was obtained and <code>user_id</code> is the user ID used to establish the connections contained in the pool. The following example shows how to return the Connection Pool Monitor for the connection pool that is bound to the JNDI lookup name <code>jdbc/PoolHybridSparky</code> and connections established by user <code>test04</code>.</td>
</tr>
<tr>
<td>boolean isTracing()</td>
<td>Determines whether tracing is enabled. If enabled, tracing information is sent to the PrintWriter that is passed to the setLogWriter() method or the standard output <code>System.out</code> if the setLogWriter() method is not called.</td>
</tr>
<tr>
<td>void setDataSourceName(String dataSourceName)</td>
<td>Sets the JNDI name, which is used to look up the driver <code>DataSource</code> object referenced by this PooledConnectionDataSource. The driver <code>DataSource</code> object bound to this PooledConnectionDataSource, specified by <code>dataSourceName</code>, is not persisted. Any changes made to the PooledConnectionDataSource bound to the specified driver <code>DataSource</code> object affect this PooledConnectionDataSource.</td>
</tr>
<tr>
<td>void setDataSourceName(String dataSourceName, ConnectionPoolDataSource dataSource)</td>
<td>Sets the JNDI name associated with this PooledConnectionDataSource, specified by <code>dataSourceName</code>, and the driver <code>DataSource</code> object, specified by <code>dataSource</code>, referenced by this PooledConnectionDataSource. The driver <code>DataSource</code> object, specified by <code>dataSource</code>, is persisted with this PooledConnectionDataSource. Changes made to the specified driver <code>DataSource</code> object after this PooledConnectionDataSource is persisted do not affect this PooledConnectionDataSource.</td>
</tr>
<tr>
<td>void setDataSourceName(String dataSourceName, Context ctx)</td>
<td>Sets the JNDI name, specified by <code>dataSourceName</code>, and context, specified by <code>ctx</code>, to be used to look up the driver <code>DataSource</code> referenced by this PooledConnectionDataSource. The JNDI name, specified by <code>dataSourceName</code>, and context, specified by <code>ctx</code>, are used to look up a driver <code>DataSource</code> object. The driver <code>DataSource</code> object is persisted with this PooledConnectionDataSource. Changes made to the driver <code>DataSource</code> after this PooledConnectionDataSource is persisted do not affect this PooledConnectionDataSource.</td>
</tr>
</tbody>
</table>
### PooledConnectionDataSource Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>void setDescription(String description)</code></td>
<td>Sets the description of the PooledConnectionDataSource, where <code>description</code> is the description.</td>
</tr>
<tr>
<td><code>void setInitialPoolSize(int initialPoolSize)</code></td>
<td>Sets the value of the initial pool size, which is the number of connections created when the connection pool is initialized.</td>
</tr>
<tr>
<td><code>void setLoginTimeout(int i)</code></td>
<td>Sets the value of the login timeout, where <code>i</code> is the login timeout, which is the time allowed for the data store login to be validated.</td>
</tr>
<tr>
<td><code>void setLogWriter(PrintWriter printWriter)</code></td>
<td>Sets the writer, where <code>printWriter</code> is the writer to which the stream will be printed.</td>
</tr>
<tr>
<td><code>void setMaxIdleTime(int maxIdleTime)</code></td>
<td>Sets the value of the maximum idle time, which is the time a connection can remain idle in the connection pool before it is closed and removed from the pool.</td>
</tr>
<tr>
<td><code>void setMaxPoolSize(int maxPoolSize)</code></td>
<td>Sets the value of the maximum pool size, which is the maximum number of connections for each user allowed in the pool. See Configuring pool size on page 820 for more information about how the Pool Manager implements the maximum pool size.</td>
</tr>
<tr>
<td><code>void setMaxPoolSizeBehavior(String value)</code></td>
<td>Sets the value of the maximum pool size behavior, which is either <code>softCap</code> or <code>hardCap</code>. If <code>setMaxPoolSizeBehavior(softCap)</code>, the number of active connections may exceed the maximum pool size, but the number of idle connections in the connection pool for each user cannot exceed this limit. If a user requests a connection and an idle connection is unavailable, the Pool Manager creates a new connection for that user. When the connection is no longer needed, it is returned to the pool. If the number of idle connections exceeds the maximum pool size, the Pool Manager closes idle connections to enforce the maximum pool size limit. This is the default behavior. If <code>setMaxPoolSizeBehavior(hardCap)</code>, the total number of active and idle connections cannot exceed the maximum pool size. Instead of creating a new connection for a connection request if an idle connection is unavailable, the Pool Manager queues the connection request until a connection is available or the request times out. This behavior is useful if your data store server has memory limitations or is licensed for only a specific number of connections. The timeout is set using the LoginTimeout connection property. If the connection request times out, the driver throws an exception. See Configuring pool size on page 820 for more information about how the Pool Manager implements the maximum pool size.</td>
</tr>
<tr>
<td><code>void setMinPoolSize(int minPoolSize)</code></td>
<td>Sets the value of the minimum pool size, which is the minimum number of idle connections to be kept in the connection pool.</td>
</tr>
</tbody>
</table>
### PooledConnectionDataSource Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>void setPropertyCycle(int propertyCycle)</td>
<td>Sets the value of the property cycle, which specifies how often the pool maintenance thread wakes up and checks the connection pool.</td>
</tr>
<tr>
<td>void setTracing(boolean value)</td>
<td>Enables or disables tracing. If set to true, tracing is enabled; if false, it is disabled. If enabled, tracing information is sent to the PrintWriter that is passed to the setLogWriter() method or the standard output System.out if the setLogWriter() method is not called.</td>
</tr>
</tbody>
</table>

### PooledConnectionDataSourceFactory

The **PooledConnectionDataSourceFactory** interface is used to create a **PooledConnectionDataSource** object from a **Reference** object that is stored in a naming or directory service. These methods are typically invoked by a JNDI service provider; they are not usually invoked by a user application.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static Object getObjectInstance(Object refObj, Name name, Context nameCtx, Hashtable env)</td>
<td>Creates a PooledConnectionDataSource object from a Reference object that is stored in a naming or directory service. This is an implementation of the method of the same name defined in the javax.naming.spi.ObjectFactory interface. Refer to the Javadoc for this interface for a description.</td>
</tr>
</tbody>
</table>

### ConnectionPoolMonitor

The **ConnectionPoolMonitor** interface is used to return information that is useful for monitoring the status of your connection pools.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getName()</td>
<td>Returns the name of the connection pool associated with the monitor. The connection pool name has the form: <code>jndi_name-user_id</code> where <code>jndi_name</code> is the name used for the JNDI lookup of the PooledConnectionDataSource object from which the pooled connection was obtained and <code>user_id</code> is the user ID used to establish the connections contained in the pool.</td>
</tr>
<tr>
<td>int getNumActive()</td>
<td>Returns the number of connections that have been checked out of the pool and are currently in use.</td>
</tr>
<tr>
<td>int getNumAvailable()</td>
<td>Returns the number of connections that are idle in the pool (available connections).</td>
</tr>
</tbody>
</table>
## Methods for configuring the connection pool

You can configure attributes of a DataDirect connection pool for optimal performance and scalability using the methods provided by the DataDirect Connection Pool Manager classes (see DataDirect Connection Pool Manager interfaces on page 815).

Some commonly set connection pool attributes include those that control pool size and idle time.

- **Minimum pool size**, which is the minimum number of connections that will be kept in the pool for each user
- **Maximum pool size**, which is the maximum number of connections in the pool for each user
- **Initial pool size**, which is the number of connections created for each user when the connection pool is initialized
- **Maximum idle time**, which is the amount of time a pooled connection remains idle before it is removed from the connection pool

### Configuring pool size

Set the maximum pool size using the `PooledConnectionDataSource.setMaxPoolSize()` method. For example, the following code sets the maximum pool size to 10 connections:

```java
ds.setMaxPoolSize(10);
```

You can control how the Pool Manager implements the maximum pool size by setting the `PooledConnectionDataSource.setMaxPoolSizeBehavior()` method:

- **If setMaxPoolSizeBehavior(softCap)**, the number of active connections can exceed the maximum pool size, but the number of idle connections for each user in the pool cannot exceed this limit. If a user requests a connection and an idle connection is unavailable, the Pool Manager creates a new connection for that user. When the connection is no longer needed, it is returned to the pool. If the number of idle connections for each user exceeds the limit, the Pool Manager may remove available connections from the pool.

<table>
<thead>
<tr>
<th>ConnectionPoolMonitor Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int getInitialPoolSize()</code></td>
<td>Returns the initial size of the connection pool (the number of available connections in the pool when the pool was first created).</td>
</tr>
<tr>
<td><code>int getMaxPoolSize()</code></td>
<td>Returns the maximum number of available connections in the connection pool. If the number of available connections exceeds this value, the Pool Manager removes one or more available connections from the pool.</td>
</tr>
<tr>
<td><code>int getMinPoolSize()</code></td>
<td>Returns the minimum number of available connections in the connection pool. When the number of available connections is lower than this value, the Pool Manager creates additional connections and makes them available.</td>
</tr>
<tr>
<td><code>int getPoolSize()</code></td>
<td>Returns the current size of the connection pool, which is the total of active connections and available connections.</td>
</tr>
</tbody>
</table>
connections exceeds the maximum pool size, the Pool Manager closes idle connections to enforce the pool size limit. This is the default behavior.

- If `setMaxPoolSizeBehavior(hardCap)`, the total number of active and idle connections cannot exceed the maximum pool size. Instead of creating a new connection for a connection request if an idle connection is unavailable, the Pool Manager queues the connection request until a connection is available or the request times out. This behavior is useful if your client or application server has memory limitations or if the data store server is licensed for only a certain number of connections.

See [PooledConnectionDataSource](#) on page 815 for more information about these methods.

## Checking the Pool Manager version

To check the version of your DataDirect Connection Pool Manager, navigate to the directory containing the DataDirect Connection Pool Manager (`install_dir/pool manager` where `install_dir` is your product installation directory). At a command prompt, enter the command:

**On Windows:**

```
java -classpath poolmgr_dir\pool.jar com.ddtek.pool.PoolManagerInfo
```

**On UNIX:**

```
java -classpath poolmgr_dir/pool.jar com.ddtek.pool.PoolManagerInfo
```

where `poolmgr_dir` is the directory containing the DataDirect Connection Pool Manager.

Alternatively, you can obtain the name and version of the DataDirect Connection Pool Manager programmatically by invoking the following static methods:

```
com.ddtek.pool.PoolManagerInfo.getPoolManagerName()
```

and

```
com.ddtek.pool.PoolManagerInfo.getPoolManagerVersion()
```

## Enabling Pool Manager tracing

You can enable Pool Manager tracing by calling `setTracing(true)` on the `PooledConnectionDataSource` connection. To disable logging, call `setTracing(false)`.

By default, the DataDirect Connection Pool Manager logs its pool activities to the standard output `System.out`. You can change where the Pool Manager trace information is written by calling the `setLogWriter()` method on the `PooledConnectionDataSource` connection.

See [Troubleshooting Connection Pooling](#) on page 752 for information about using a Pool Manager trace file for troubleshooting.
JDBC extensions

This section describes the JDBC extensions provided by the `com.ddtek.jdbc.extensions` package. Your application can take advantage of these extensions.

JDBC Wrapper methods to access JDBC extensions

The Wrapper methods allow an application to access vendor-specific classes. The following example shows how to access the DataDirect-specific `ExtConnection` class using the Wrapper methods:

```java
ExtStatementPoolMonitor monitor = null;
Class<ExtConnection> cls = ExtConnection.class;
if (con.isWrapperFor(cls)) {
    ExtConnection extCon = con.unwrap(cls);
    extCon.setClientUser("Joe Smith");
    monitor = extCon.getStatementPoolMonitor();
}...
if (monitor != null) {
    long hits = monitor.getHitCount();
    long misses = monitor.getMissCount();
}...
```

ExtConnection interface

Table 143: Methods of the ExtConnection Interface

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ExtStatementPoolMonitor</code></td>
<td>Returns an ExtStatementPoolMonitor object for the statement pool associated with the connection. If the connection does not have a statement pool, this method returns null.</td>
</tr>
<tr>
<td><code>getStatementPoolMonitor()</code></td>
<td></td>
</tr>
</tbody>
</table>

SQL escape sequences

JDBC defines escape sequences that contain the standard syntax for the following language features:

- Date, time, and timestamp literals
- Scalar functions such as numeric, string, and data type conversion functions
- Outer joins
- Escape characters for wildcards used in LIKE clauses
- Procedure calls

The escape sequence used by JDBC is:

```
{extension}
```
The escape sequence is recognized and parsed by the driver, which replaces the escape sequences with data store-specific grammar.

**Date, Time, and Timestamp escape sequences**

**Syntax**

\[ \text{\{} \text{literal-type} \ 'value' \ \text{\}} \]

where:

- **literal-type** is one of the following:

<table>
<thead>
<tr>
<th>Literal-type</th>
<th>Description</th>
<th>Value Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>d</td>
<td>Date</td>
<td>yyyy-mm-dd</td>
</tr>
<tr>
<td>t</td>
<td>Time</td>
<td>hh:mm:ss []</td>
</tr>
<tr>
<td>ts</td>
<td>Timestamp</td>
<td>yyyy-mm-dd hh:mm:ss [.f...]</td>
</tr>
</tbody>
</table>

**Example**

```
UPDATE Orders SET OpenDate={d '1995-01-15'} WHERE OrderID=1023
```

**Scalar functions**

Scalar functions are specific to each type of data store. Refer to the documentation for the data source to which you are connection.

The driver supports a variety of scalar functions, which return a single value based on the input value. The SQLGetInfo function returns information about supported functions.

Applications can construct SQL statements using the following syntax:

```
{fn scalar-function}
```

For example:

```
SELECT {fn UCASE(NAME)} FROM EMP
```

Applications connecting through JDBC can use the following scalar functions in expressions. For syntax details, consult your JDBC documentation.

**Outer join escape sequences**

JDBC supports the SQL92 left, right, and full outer join syntax.
Syntax

{oj table-reference {LEFT | RIGHT | FULL} OUTER JOIN {table-reference | outer-join} ON search-condition}

where:

table-reference

is a table name.

search-condition

is the join condition you want to use for the tables.

Example

SELECT Customers.CustID, Customers.Name, Orders.OrderID, Orders.Status
FROM {oj Customers LEFT OUTER JOIN
Orders ON Customers.CustID=Orders.CustID}
WHERE Orders.Status='OPEN'

LIKE escape character sequence for wildcards

You can specify the character to be used to escape wildcard characters (% and _, for example) in LIKE clauses.

Syntax

{escape 'escape-character'}

where:

escape-character

is the character used to escape the wildcard character.

Example

The following SQL statement specifies that an asterisk (*) be used as the escape character in the LIKE clause for the wildcard character %:

SELECT col1 FROM table1 WHERE col1 LIKE '%%% {escape '*'}

Procedure call escape sequences

A procedure is an executable object stored in the data store. Generally, it is one or more SQL statements that have been precompiled.

Syntax

{{?=}call procedure-name[({parameter[,{parameter}...})]}}

where:
procedure-name

is the name of a stored procedure.

parameter

is a stored procedure parameter.
Querying with OData Version 2

For details, see the following topics:

• Getting started with OData Version 2
• Supported functionality for OData Version 2
• Understanding and configuring a schema map for OData Version 2
• Structure of requests for OData Version 2
• Formulating queries with OData Version 2
• Method Reference for OData Version 2

Getting started with OData Version 2

This section describes using Hybrid Data Pipeline to query data with OData Version 2. Hybrid Data Pipeline also supports OData Version 4. For information on querying with OData Version 4, see Getting started with OData Version 4 on page 863.

The Open Data Protocol (OData) provides a standard for exposing resources using Uniform Resource Identifiers (URIs) and an API for querying the resources with simple HTTP messages. Hybrid Data Pipeline OData services support OData requests for a variety of data stores. Since OData is REST-based, and does not require any locally-installed software, the Hybrid Data Pipeline OData API provides quick and easy data access for mobile apps and desktop applications.
The OData API is based on an object model instead of the tabular representation used by many data stores. To translate OData requests, Hybrid Data Pipeline requires a schema map. As part of a data source definition, you use the Configure Schema editor to select the tables (or objects) and columns (or attributes) to access with OData. Hybrid Data Pipeline generates a JSON schema map that exposes your selections as entities and their properties.

**Using OData**

To access a data store using OData requires both Hybrid Data Pipeline configuration and implementation on the client-side.

1. While logged into Hybrid Data Pipeline, create or edit a data source definition.

2. In the data source definition, enable OData access by Configuring data sources for OData Version 2 connectivity on page 623.

3. In the client, create requests to the OData-enabled data source, as demonstrated in Testing data source configurations (OData Version 2) on page 833 and described in more detail in Formulating queries with OData Version 2 on page 846.

**Configuring data sources for OData Version 2 connectivity**

Hybrid Data Pipeline supports OData Version 2 and Version 4 connectivity for all supported data stores. You can configure a data source on any data store for OData connectivity either during the process of creating the data source or after the data source has been created.

The following steps describe how to configure a data source for OData Version 2 connectivity.

1. From the Web UI, navigate to the Data Sources view by clicking the data sources icon.
   - **Option 1.** If creating a new data source, click New Data Source, choose the data store, enter the required information on the General tab, and click TEST to confirm connectivity to the backend data store. (See Creating data sources with the Web UI on page 224 for details.)
   - **Option 2.** If enabling OData on an existing data source, select the data source you wish to modify.

2. Select the OData tab.
3. For **OData Version**, select **Version 2**.

4. Open the Configure Schema editor by clicking **Configure** to the right of the **Schema Map** field.

5. Select a schema from the **Select Schema** dropdown.

   **Note:** By default, Hybrid Data Pipeline exposes all schemas on any backend data stores that support multiple schemas. The **Metadata Exposed Schemas** option on the **Advanced** tab for any such data store can be used to limit exposed schemas to a single schema. If a schema is selected for the **Metadata Exposed Schemas** option, it will be the only schema available on the Configure Schema editor's **Select Schema** dropdown.

6. Select the **Tables and Columns** tab. Then select and define the tables and columns you want to expose to OData client applications.
• To add all tables, click **Add All Tables** on the **Tables** panel.

• To add individual tables, select a table on the **Tables** panel and click **Add To Map** in the **Settings** panel to the right.

• To remove a table that was previously added, select the table and click **Remove From Map** in the **Settings** panel.

• To specify singular and plural alias names for a table, select the table, enter the table alias for the entity type name in the **Singular Name** field, enter the table alias for the entity collection name in the **Plural Name** field, and click **Add To Map**.

**Note:** The singular alias name specified is used as the entity type name, while the plural alias name will be used as the entity collection name. If alias names are not specified, the table name is used as the entity type name and pluralized for the entity collection name. For example, the entity type name for the table **ACCOUNTS** would be **ACCOUNTS**, while the entity collection name would be **ACCOUNTSES**.

• To specify a column as a primary key, select the column from the **Columns** panel and set the **Is Primary Key** switch from **OFF** to **ON**.

**Note:** The Configure Schema editor indicates that a primary key exists for a table with a skeleton key icon. A primary key assigned in the backend data store cannot be changed. If a primary key has not been discovered for a table you wish to map, one or more columns must be specified as a primary key.

• To remove a column from the OData schema map, select the column from the **Columns** panel and click **Remove From Map** in the **Settings** panel.

**Note:** When a table is added, all columns in the table are exposed in the OData schema map by default. You can modify the columns exposed by removing (or excluding) them from the schema map.

7. Take the following steps to enable text search for individual tables and text-based columns using the `ddsearch` custom query parameter.

   a) Select a table from the **Tables** panel.

   b) Specify a search option from the **Search Options** dropdown. Then click **Add To Map**.

   • **Full Text** is only available for data store types that support indexing and full text search.

   • **Substring** enables searches for the string anywhere in the search-enabled fields.

   • **Begins** restricts the search to the text at the beginning of a field.

   c) If you selected **Full Text** in Step b, you should select an index type for all text-based columns. Select the column from the **Columns** panel, and specify an index type from the **Index Type** dropdown in the **Settings** panel. Then click **Add To Map**.
The index type is the type of index supported by the backend data store. **TEXT** is the only valid value for the DB2 and SQL Server data stores. **CONTEXT** and **CTXCAT** are the valid values for the Oracle data store. If **Full Text** has been selected but the data store index has not been properly configured, queries using ddsearch will return errors.

d) If you selected **Substring** or **Begins** in Step b, you should select which text-based columns can be searched. Select the column from the **Columns** panel, and set the **Is Searchable** switch to **ON**. Then click **Add To Map**.

8. Click the **Review Schema Map** tab to review the OData schema map in JSON format.

9. Click **Save Map** to save your configuration of the OData schema map.

10. Set OData options to the desired values.

- **Data Source Caching** controls caching of results. A value of 0 results in a stateless session, requiring Hybrid Data Pipeline to access the data store for each request. A value of 1, the default, allows Hybrid Data Pipeline to cache results, improving performance for repeated requests for the same entity. Hybrid Data Pipeline clears the cache after ten minutes of inactivity or at the end of a session.

- **Page Size** controls the number of results returned in one response. By default, the value in this field is 0 which causes Hybrid Data Pipeline to return up to 2,000 top-level entities per response. If the response contains more than 2,000 entities, the first 2,000 entities are returned and the end of the response contains a link that the OData client can use to fetch the next set. You can set the page size by using values from 1 to 10,000. Client requests can also specify the size of results with query parameters.

- **Refresh Result** determines whether Hybrid Data Pipeline returns results from the cache (for entities in the cache) or queries the data source again. A value of 1, the default, allows Hybrid Data Pipeline to satisfy requests from cached results. A value of 0 forces queries to the backend data store. If caching is not enabled, this parameter has no effect.

- **Inline Count Mode** controls how Hybrid Data Pipeline handles requests that include the $inlinecount parameter with a value of allpages. The response includes the total number of entities that satisfy the query. A value of 0 causes Hybrid Data Pipeline to skip counting. A value of 1 causes Hybrid Data Pipeline to run a separate query to get the count before the query that returns the entities. This can result in the first page of results being returned faster for large result sets for some data store types. A value of 2, the default, causes Hybrid Data Pipeline to fetch all results and calculate the total number before returning the first page of results to the client.
• **Top Mode** allows Hybrid Data Pipeline to better handle requests that include the `$top` parameter. A value of 0, the default, indicates that clients using `$top` to limit result set size will rarely attempt to get additional entities using the `$skip` parameter. A value of 1 indicates that clients generally use `$top` and `$skip` together to paginate results.

• **OData Read Only** controls read/write access. For a new data source definition, this option is not selected by default. For a data source definition where OData was enabled before this option was available, it will be checked by default. Remove the check mark to enable write access.

11. Click **Update** to save your work.

**What to do next:**

Test your OData-enabled data source as described in [Testing data source configurations (OData Version 2)](page833) on page 833.

After you create an OData-enabled data source, you can view the status of the schema map generation on the **Data Sources** screen. The icon besides the OData-enabled data source indicates the status of the schema map generation. The following table provides details of the icons.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Progress" /></td>
<td>The synchronization of the schema map is in progress. The number denotes the percentage of synchronization completed.</td>
</tr>
<tr>
<td><img src="image" alt="Success" /></td>
<td>The schema map was synchronized successfully.</td>
</tr>
<tr>
<td><img src="image" alt="Warning" /></td>
<td>The schema map was synchronized successfully, but there are some table/column warnings. Hybrid Data Pipeline allows users to know the details of the tables/columns and/or functions that were dropped while generating the OData Model for a given schema map of a Data Source. The number of warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the <a href="page1412">Schema API</a> on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td><img src="image" alt="Error" /></td>
<td>Errors occurred while synchronizing the schema map. You must address the errors and synchronize the schema map again. Hybrid Data Pipeline allows users to know the details of the tables and/or columns that were dropped while generating the OData Model for a given schema map of a Data Source. The number of errors/warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the <a href="page1412">Schema API</a> on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td><img src="image" alt="Refresh" /></td>
<td>You must synchronize the schema map again.</td>
</tr>
</tbody>
</table>
Testing data source configurations (OData Version 2)

You can quickly test the configuration from the Hybrid Data Pipeline dashboard or by using a REST client, as described below.

- Testing data source configurations from the Hybrid Data Pipeline dashboard on page 833.
- Testing data source configurations using a REST client on page 833

Testing data source configurations from the Hybrid Data Pipeline dashboard

To test whether your data source definition and schema map are configured correctly:

1. In the left navigation pane, select Data Sources to open your list of data sources.
2. Select the OData-enabled data source definition, and click the OData URI icon at the end of the row.
3. Enter your Hybrid Data Pipeline credentials.

   The browser returns an XML document listing the entities in the schema.

Testing data source configurations using a REST client

Take the following steps to test a data source configuration using a REST client. In this example, Postman is used as the REST client.

1. Using the controls exposed by the REST client, select basic authorization and enter your Hybrid Data Pipeline credentials.
2. If credentials for your data store are not saved in the data source definition, pass them as values for ddcloud-datasource-user and ddcloud-datasource-password headers.
3. From the OData tab of the data source you are testing, copy the OData Access URI. Then paste the URI in the URL field of the REST client.
4. Execute a GET on the data source endpoint. For example:

   GET https://service.myserver.com/api/odata/db2ds

The response payload returns a list of entities exposed by the OData schema map.
Requesting service metadata and the service document

Metadata for an OData service can be fetched by requesting the service document or service metadata using a GET request.

**Service Document**

The service document returns a list of all the available entities in a schema in the request payload. To fetch the service document, issue a GET request for the data source's service root.

```
<server>:<port>/api/odata/<hdp_data_source>
```

For example:

https://MyServer:8443/api/odata/myds/

**Service Metadata**

Fetching service metadata returns a description of the data model for the service, including the names, properties, data types, and relationships for all entities in the schema. To fetch service metadata, issue a GET request for the data source's service root with `/$metadata` appended to the path:

```
<server>:<port>/hdp_data_source/$metadata
```

For example:

https://MyServer:8443/api/odata/myds/$metadata
Supported functionality for OData Version 2

Hybrid Data Pipeline supports the OData Version 4.0 and Version 2.0 specifications.

Data sources and data source groups support using a single supported version of the specification at a time. The version used by a data source is determined by the setting of the OData Version parameter on the OData tab. The OData version of a data source group must match the OData version of its member data sources.

This section describes using Hybrid Data Pipeline with OData Version 2. For information on using Hybrid Data Pipeline with OData Version 4, see Getting started with OData Version 4 on page 863.

Supported OData operations and data types

Supported OData API operations

The following table shows the operations that can be performed and their associated URLs. Query the data source name to get a list of the valid entities.

In the URL examples in this table,

<myserver> is the DNS name or the IP address of the machine on which Hybrid Data Pipeline is installed.

<myds> is the name of your Hybrid Data Pipeline data source.

<plural-name> is the name you designate in your schema map for entity plurals. In the schema map, Hybrid Data Pipeline pluralizes the table name automatically. You use the plural entity name in OData requests.

pkey is the primary key.
<table>
<thead>
<tr>
<th>Purpose</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fetch Data from an OData Service</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:8443/api/odata/&lt;myds&gt;/&lt;plural-name&gt;</td>
</tr>
<tr>
<td>Create an Entity</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:8443/api/odata/&lt;myds&gt;/&lt;plural-name&gt;</td>
</tr>
<tr>
<td>Update an Entity</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:8443/api/odata/&lt;myds&gt;/&lt;plural-name&gt;('pkey')</td>
</tr>
<tr>
<td>X-HTTP-Method:MERGE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delete an Entity</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:8443/api/odata/&lt;myds&gt;/&lt;plural-name&gt;('pkey')</td>
</tr>
<tr>
<td>Or</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:8443/api/odata/&lt;myds&gt;/&lt;plural-name&gt;('pkey')</td>
</tr>
<tr>
<td>X-HTTP-Method:DELETE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Entity Data Model (EDM) types for OData Version 2**
To support communication between an OData client and a backend data store, Hybrid Data Pipeline uses a schema map to convert data to the appropriate type for the receiver. You configure the schema map in Hybrid Data Pipeline where it is generated as a JSON string with the following OData Entity Data Model (EDM) types.

### Table 144: Supported Data Types for OData version 2

<table>
<thead>
<tr>
<th>SQL Data Type</th>
<th>EDM Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>Edm.Int64</td>
</tr>
<tr>
<td>BINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>BIT</td>
<td>Edm.Boolean</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>Edm.Boolean</td>
</tr>
<tr>
<td>CHAR</td>
<td>Edm.String</td>
</tr>
<tr>
<td>DATE</td>
<td>Edm.DateTime</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>Edm.Decimal</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>FLOAT</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Edm.Int32</td>
</tr>
<tr>
<td>LONGVARBINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>LONGVARCHAR</td>
<td>Edm.String</td>
</tr>
<tr>
<td>REAL</td>
<td>Edm.Single</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>Edm.Int16</td>
</tr>
<tr>
<td>TIME</td>
<td>Edm.DateTime</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>Edm.DateTime (no timezone)</td>
</tr>
<tr>
<td></td>
<td>Edm.DateTimeOffset (with timezone)</td>
</tr>
<tr>
<td>TINYINT</td>
<td>Edm.SByte</td>
</tr>
<tr>
<td>VARBINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>Edm.String</td>
</tr>
</tbody>
</table>

1For values smaller than 32 KB. Values 32 KB and larger are not supported.
Understanding and configuring a schema map for OData Version 2

As described in Configuring data sources for OData Version 2 connectivity on page 623, you use the Hybrid Data Pipeline dashboard's Configure Schema editor to generate or edit a schema map. The schema map specifies the tables, or objects, and columns that will be accessible to OData clients for a particular data source definition. A schema map can only include tables from one schema. To expose tables from multiple schemas (in the same data store) or to expose multiple data stores in a single OData endpoint, you can create a data source group.

Hybrid Data Pipeline generates schema maps as a JSON string. When fetching data to satisfy requests, the Hybrid Data Pipeline OData service uses this schema to map a row in a table (or an object instance) to an entity, and to map the data in table columns (or object attributes) to entity properties. Progress recommends that you use the generated schema map. However, there are rare use cases that might require you to edit the JSON string. See JSON schema map syntax on page 839 for a description of the syntax.

Primary and foreign keys

The schema map must specify how to uniquely identify a particular record. Many data store tables already have one or more primary key columns. The Configure Schema editor checks for a primary key in the tables you select, and identifies all tables that need to have a primary key defined. If a primary key is defined on a table, the OData service uses that primary key as the unique identifier and you cannot specify another. To expose tables that do not have a primary key, you must choose one or more columns to use as a virtual primary key. Hybrid Data Pipeline automatically adds related tables for selected foreign key columns.

Note: Although the Configure Schema editor lets you specify which tables and columns to expose to OData requests, it makes no change in the underlying data source. All columns of the data source are still available to SQL queries executed from the ODBC driver, JDBC driver, or the Hybrid Data Pipeline SQL Editor regardless of whether they are exposed through OData.

Entity names

In some cases, you might want to modify the names that the Configure Schema editor assigns to an entity.

- By default, the Hybrid Data Pipeline OData service uses a plural form of the table name as the entity name. The schema generator automatically appends es to table names. For example, a data source table named Customers will become a Customeres entity. You might want to explicitly set the name to Customers.
- If you are using a data source group, table names in the member data sources can conflict. Therefore, when you create a data source group, you must assign a unique prefix to each data source definition. When this is the case, it makes sense to use the same plural name for the tables in each schema map.

Queries must have the prefix appended to the plural entity name with an underscore separator. For example, two data sources in the same group might contain a Customer table. In the Configure Schema editor, you could assign the plural name Customers to the tables in both schemas. In the data source group, you could use a prefix such as east for one member and west for the other. Query requests to the east_Customers entity will then go to the first data source, and requests to west_Customers to the second.
JSON schema map syntax

The Configure Schema editor should be used to generate the OData schema map as described in Configuring data sources for OData Version 2 connectivity on page 623. In rare cases, manual editing of the schema map might be necessary. For OData Version 2 services, an odata_mapping_v2 format is supported.

A schema map consists of a JSON string that contains the following model elements:

```json
{
   "odata_mapping_v2": {
       "schemas": [
           {
               "name": "<schema_name>",
               "tables": {
                   "<table_name>": {
                       "ODataAlias": "<odata_name>",
                       "ODataPluralAlias": "<plural_odata_name>",
                       "searchMode": "none" or "begins" or "contains" or "full-text",
                       "columns": {
                           "<column_name>": {
                               "primaryKeyComponent": «integer»,
                               "searchable": «boolean»,
                               "indexType": "<text_index_name>"
                           }, ...
                       },
                       "excludedColumns": ["<column_name>", ...]
                   }, ...
               },
               "excludedTables": ["<table_name>", ...]
           }, ...
       }
   }
}
```

The following table lists elements alphabetically and provides a brief description. See Schema map examples on page 841 for sample usage.

<table>
<thead>
<tr>
<th>Element name</th>
<th>Parent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>columns</td>
<td>table_name</td>
<td>Contains column_name elements that define the details of columns included in a table. If the columns element is missing or empty, then all columns except the ones listed in excludeColumns are exposed.</td>
</tr>
<tr>
<td>column_name</td>
<td>columns</td>
<td>Backend data source column (or field) name. Properties determine whether the column is part of the primary key and is searchable.</td>
</tr>
<tr>
<td>excludedColumns</td>
<td>table_name</td>
<td>Comma-separated list of columns to hide from OData requests. This optional field is used only when the columns object is missing or empty.</td>
</tr>
<tr>
<td>excludedTables</td>
<td>schema_name</td>
<td>Comma-separated list of tables to hide from OData requests. Any tables not specified in this list that have a primary key column will be exposed for OData requests. This optional field is used only when the tables object is missing or empty.</td>
</tr>
<tr>
<td>Element name</td>
<td>Parent</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>indexType</td>
<td>column_name</td>
<td>The model contains this element to identify the type of index when the search mode is set to Full Text. For DB2 and SQL Server, TEXT is the only valid value. For Oracle, valid values include CONTEXT and CTXCAT.</td>
</tr>
<tr>
<td>name</td>
<td>schema_name</td>
<td>Contains the schema_name element. This is a required property for data sources that support schemas. For data sources such as MySQL that do not support schemas, set this to &quot;null&quot; or &quot;.&quot;.</td>
</tr>
<tr>
<td>ODataAlias</td>
<td>table_name</td>
<td>The singular entity name to use in OData addresses for requests to this table.</td>
</tr>
<tr>
<td>ODataPluralAlias</td>
<td>table_name</td>
<td>The plural entity name to use in OData requests.</td>
</tr>
<tr>
<td>primaryKeyComponent</td>
<td>column_name</td>
<td>The data type of a column belonging to the primary key or null. The primary key is comprised of a set of columns to use as the primary key for a table that does not have a defined primary key. If this field is not specified or the key list is empty, the table must have a primary key defined in the database. If a primary key is defined for the table in the database and a primary key column list is also specified in the OData Schema Map parameter, the primary key defined in the database is used.</td>
</tr>
<tr>
<td>schema_name</td>
<td>None</td>
<td>Backend data source schema name, a required field. For data stores that do not support schemas, such as MySQL, the schemaName value should be null (&quot;schemaName&quot;: null).</td>
</tr>
<tr>
<td>searchable</td>
<td>column_name</td>
<td>If true, the column is searchable, using the searchMode specified at the table level. If false, the column is not searchable.</td>
</tr>
<tr>
<td>searchMode</td>
<td>table_name</td>
<td>One of: none, not searchable; begins, search for the string only at the beginning of a field; contains, search for a specific string; full-text, use the data source index. The searchMode applies to columns enabled for search.</td>
</tr>
<tr>
<td>table_name</td>
<td>tables</td>
<td>Backend data source table (or object) name. Properties determine the name to be used in OData requests, primary key column(s), and whether any columns are searchable.</td>
</tr>
<tr>
<td>tables</td>
<td>schema_name</td>
<td>Contains table_name elements describing how to expose tables through OData, and an excludedTables element listing tables that should not be exposed. If the tables object is missing or empty, all tables, except for any table in the excludedTables array, are exposed.</td>
</tr>
</tbody>
</table>
Schema map examples

In the following example from an Oracle data source, both the Employees and the Departments tables are enabled for full-text search. In the Employees table, the id column is searchable. In the Departments table, the id column is searchable and the address column is not included in the model; OData requests will not return data from the address column.

```
{
  "odata_mapping_v2": {
    "schemas": [
      {
        "name": "Emp",
        "tables": {
          "Employees": {
            "ODataAlias": "Employee",
            "ODataPluralAlias": "Employees",
            "searchMode": "full-text",
            "columns": {
              "id": {
                "primaryKeyComponent": 1
              },
              "EmployeeName": {
                "searchable": true
              }
            }
          },
          "Departments": {
            "ODataAlias": "Department",
            "searchMode": "full-text",
            "columns": {
              "id": {
                "primaryKeyComponent": 1
              },
              "DepartmentName": {
                "searchable": true
              }
            },
            "excludedColumns": ["address"]
          }
        }
      }
    ]
  }
}
```

The following example uses tables in a MySQL datasource. As in the previous example, both the Employees and the Departments tables are enabled for full-text search. In the Employees table, the id column is searchable. In the Departments table, the id column is searchable and the address column is not included in the model; OData requests will not return data from the address column.

```
{
  "odata_mapping_v2": {
    "schemas": [
      {
        "name": ",",
        "tables": {
          "Employees": {
            "ODataAlias": "Employee",
            "ODataPluralAlias": "Employees",
            "searchMode": "full-text",
            "columns": {
              "id": {
                "primaryKeyComponent": 1
              }
            }
          }
        }
      }
    ]
  }
}
```
Structure of requests for OData Version 2

OData requests to a Hybrid Data Pipeline data source must include authentication, the service root, and resource name. You can fetch single or multiple entities and related entities using entity addressing and the supported methods. While you can set some server-side behavior such as caching and paging in the data source definition, client-side options also allow you to control behaviors such as paging and response formatting.

The following are required:

• **Authentication**
  
  Supply credentials for Hybrid Data Pipeline and for the backend data store:

  • The Hybrid Data Pipeline user ID and password must be passed using HTTP basic authentication. The client encrypts the Hybrid Data Pipeline user ID and password in the Authorization header.

  • The credentials for your data store can be stored in the data source definition or passed as part of an OData request — using the `ddcloud-datasource-user` and the `ddcloud-datasource-password` headers, as described in Data Source User Header on page 844 and Data Source Password Header on page 844.

• **Service root and resource name**
  
  The location of the Hybrid Data Pipeline service and the name of the OData-enabled data source definition (case insensitive) as displayed on the OData tab of your data source definition. See Service URI and resource path on page 845 for an example.

The following are optional:

• **Entity addressing**
Append entity addresses to the request after the data source name. Use the plural entity name defined in the schema map. For example, the following request fetches the employee record with a primary key of 27, from the EMPLOYEES table in the myoracletest2 data source.

https://<myserver>:<port>/api/odata/myoracletest2/EMPLOYEES('27')

See Service URI and resource path on page 845 and Formulating queries with OData Version 2 on page 846 for details and more examples.

where <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

Note: Unless the ports 80 and 443 are redirected to 8080 and 8443 respectively, you must specify <myserver>:<port>.

- **Queries and operations**

  Hybrid Data Pipeline supports OData edit, create, update and delete operations, see examples in the Formulating queries with OData Version 2 on page 846 section.

**Headers**

You can use request headers to control the following service behaviors:

- Whether the response comes from cached data (if available) or from the back-end data store, as described in Refresh Result Header on page 843.

- Specify the backend data store credentials as described in Data Source User Header on page 844 and Data Source Password Header on page 844.

- The time zone to apply to DateTime values, see Timezone Header on page 844.

- Anticipate how clients will be the $top system query parameter with the Top Mode on page 844 to improve performance.

- How the service breaks up a result set into multiple responses with the OData Prefer Header - Max Page Size on page 845.

Some of these behaviors can be controlled with query parameters instead of in headers. See Custom query parameters on page 849.

**Refresh Result Header**

Hybrid Data Pipeline buffers the results of an OData query, allowing clients to page back and forth through the results using the $top and $skip system query parameters. The $top parameter specifies how many results to return in the first response and $skip specifies where to start in the result set to return the next set of results. When the Hybrid Data Pipeline service receives an OData query for which it has a buffered result and the $skip query parameter is either not specified or is set to zero, Hybrid Data Pipeline can page back to the beginning of the buffered result or execute a new query.

By default, Hybrid Data Pipeline treats a query where $skip is missing or set to zero as a request to re-execute the query in the backend data source. You can change default behavior in the data source definition, or in the request with the ddcloud-refresh-result header. The header value overrides the setting in the Refresh Result field of the data source definition.
<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-refresh-result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted Values</td>
<td>0, reuse cached results. 1, discard cached results and query the data store again.</td>
</tr>
<tr>
<td>Default when not specified</td>
<td>1. The service executes the query anew.</td>
</tr>
</tbody>
</table>

**Data Source User Header**
The credentials for the backend data source can be stored in the data source definition on the **General** tab. If they are not, you must supply them in requests using the **ddcloud-datasource-user** header.

<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-datasource-user</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default when not specified</td>
<td>The Hybrid Data Pipeline service checks the data source definition for this value.</td>
</tr>
</tbody>
</table>

**Data Source Password Header**
The credentials for the backend data source can be stored in the data source definition on the **General** tab. If they are not, you must supply them in requests using the **ddcloud-datasource-password** header.

<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-datasource-password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default when not specified</td>
<td>The Hybrid Data Pipeline service checks the data source definition for this value.</td>
</tr>
</tbody>
</table>

**Timezone Header**
To correctly process **DateTime** data types for clients in a different timezone than the data store, use the **ddcloud-timezone** header.

<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-timezone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted Values</td>
<td>A Java timezone id string.</td>
</tr>
<tr>
<td>Default when not specified</td>
<td>The timezone is taken from URL; GMT is used if timezone is not specified as a header or URL parameter</td>
</tr>
</tbody>
</table>

**Top Mode**
In some cases, the Hybrid Data Pipeline OData service can optimize requests to the backend data store when you use the **ddcloud-top-mode** to specify how a client will be using the $top system parameter to page through results. A value of 0 indicates that the client will use $top to limit the result set and will rarely request the remaining entities. A value of 1 indicates that the client will often use $top and $skip to page through results.

Hybrid Data Pipeline applies the optimization only to queries that meet the following conditions:

- Include a value for $top
- Do not include $skip or include $skip with a value of 0
- Do not include $expand
- Do not include $inlinecount=allpages with the inline count mode set to 2, which causes a fetch of all rows
When the conditions are met, Hybrid Data Pipeline will generate only a `SELECT` statement that includes the data store-specific syntax for limiting the rows returned. If the client queries the same entity collection again but specifies `$top` and `$skip` to fetch more entities, the service executes a new query. The results might contain some of the entities already received from the first request.

In the following example, the `ddcloud-top-mode` is set to 1, directing the Hybrid Data Pipeline service to fetch the complete result set and not to attempt optimization:

```
ddcloud-top-mode=1
```

<table>
<thead>
<tr>
<th>Name</th>
<th><code>ddcloud-top-mode</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted Values</td>
<td>0 indicates that the client will use <code>$top</code> to limit the result set and will rarely request the remaining entities. 1 indicates that the client will often use <code>$top</code> and <code>$skip</code> to page through results.</td>
</tr>
<tr>
<td>Default when not specified</td>
<td>0</td>
</tr>
</tbody>
</table>

### OData Prefer Header - Max Page Size

The OData 4.0 specification defines a Prefer header, `odata.maxpagesize`, that can be used to control the page size for server-driven paging. In server-driven paging, the server returns partial results and includes a link the client can use to get the next set of results. Hybrid Data Pipeline supports the OData 2.0 standard, but uses the `odata.maxpagesize` Prefer header from the OData 4.0 specification to control the page size for server-driven paging.

You can set the page size in the data source definition, on the **OData** tab, in the **Page Size** field. The request header value for `odata.maxpagesize` overrides the value specified in the data source definition. In the following example, the maximum page size is set to 4000, resulting in up to 4000 entities per page.

```
Prefer: odata.maxpagesize=4000
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Prefer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted Values</td>
<td><code>odata.maxpagesize=x</code> where <code>x</code> is the maximum number of top-level entities that are returned on a page.</td>
</tr>
<tr>
<td>Default when not specified</td>
<td>The page size from either the data source or the service default page size.</td>
</tr>
</tbody>
</table>

### Service URI and resource path

The service root and resource path of a request define the location of the Hybrid Data Pipeline service and the name of the OData-enabled data source definition (case insensitive). The **OData** tab of your data source definition provides this value in the **OData Access URI** field.

In the URL examples in this table, `<myserver>` is the DNS name of the machine on which the Hybrid Data Pipeline server is installed. `<myds>` is the name of your Hybrid Data Pipeline data source.

A request with just the Service URI and resource path returns a list of available entities, the `$metadata` parameter returns metadata on those entities, and an address that includes the plural entity name and a primary key value returns a single entity, as shown in the following examples:
### Response formatting

The OData specification allows a service to return responses in several different formats. The Hybrid Data Pipeline service supports Atom Pub and JSON. By default, Hybrid Data Pipeline returns responses in Atom Pub format. Requests can override this by specifying JSON format responses in one of the following ways:

- The `$format=json` query parameter.
- An `Accept` header with a value of `application/json`.

An OData request can either use the header or the query parameter; it cannot specify both.

### Formulating queries with OData Version 2

Hybrid Data Pipeline supports the following:

- A set of OData system query options and custom options to control service behavior and control result pagination.
- HTTP methods for:
  - Fetching records using `GET`
  - Creating records using `POST`
  - Updating records using `POST with the custom X-HTTP-Method MERGE`
  - Deleting records using `POST with the custom X-HTTP-Method DELETE`
• Search for text-based columns

**Query options and optimizing response times**

You can refine query results using system query options, which begin with the $ character. Add system query options to the URL to control the amount and order of data in the response. Custom query parameters on page 849 lists additional parameters specific to Hybrid Data Pipeline. In addition, topics in this section describe settings to optimize response times when using the $inlineCount system parameter or when paging through result sets.

The following table lists the OData query string options that Hybrid Data Pipeline supports. For detailed information about the system query options, refer to the OData specification.

**Table 145: Supported system query options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Support in Hybrid Data Pipeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>$expand</td>
<td>In addition to retrieving a record or collection, retrieve related records.</td>
<td>At present, supports expanding one level deep.</td>
</tr>
<tr>
<td>$filter</td>
<td>An expression or function that must evaluate to true for records that will be included in the response.</td>
<td>Supports all functionality except the isof scalar function.</td>
</tr>
<tr>
<td>$inlinecount</td>
<td>Include a count of the number of Entries in the response. The count will be calculated after applying any $filter System Query Options present in the URI. See Improving performance when using inlineCount on page 848 for more information.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td>$orderby</td>
<td>Determines the values used to order a collection of records.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td>$count</td>
<td>Returns the number of records in a collection, or if the collection has a filter, the number of records that match the filter.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td>$value</td>
<td>Gets the raw value of a property.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td>$top</td>
<td>Identifies a subset of records to return from a collection. To form this subset, select only the first N items of the set, where N is a positive integer. See Paging through results on page 848 for more information.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td>$skip</td>
<td>Identifies a subset of records to return from a collection. Define the subset by seeking N Entries into the Collection and selecting only the remaining Entries (starting with Entry N+1), where N is a positive integer.</td>
<td>Supports all standard functionality.</td>
</tr>
</tbody>
</table>
Improving performance when using inlineCount

The $inlinecount OData system query option includes the count of the number of entities that satisfy a query in the response. The count is included in the first page in server-side paging, and in every page when the client controls paging. Possible values for the parameter include allpages and none:

inlineCountQueryOp = "$inlinecount=" "allpages" | "none"

Calculating the count for very large collections can take time. The default behavior for Hybrid Data Pipeline differs for relational and cloud data sources:

- For relational data stores, by default, Hybrid Data Pipeline sends a separate query to get the count before requesting the records. This behavior tends to result in a quicker response for the first page of results. However, it requires two queries to be executed rather than one. And, in some data sources, the count(*) aggregate is not efficiently implemented.

- For cloud-based data stores, by default, Hybrid Data Pipeline fetches the entire result before returning the first page. For small results, this approach will always be faster. However, this approach may have longer initial response time for the first page if the result is large.

This behavior can be changed in the data source definition, as described in Configuring data sources for OData Version 2 connectivity on page 623 or by using the $inlineCount parameter. With a value of allpages, Hybrid Data Pipeline will include the count in the response. For example:

https://<myserver>:<port>/api/odata/OracleOPTest/CUSTOMERS?$inlinecount=allpages

With a value of none, Hybrid Data Pipeline avoids obtaining a count and avoiding the associated overhead. For example:

https://<myserver>:<port>/api/odata/OracleOPTest/CUSTOMERS?$inlinecount=none

Paging through results

Hybrid Data Pipeline divides results that exceed a threshold into multiple pages. For OData queries, you can use server-side or client-side pagination:

- By default, Hybrid Data Pipeline divides OData responses with a maximum of 2000 top-level entities per response. If the response is larger than 2000 entities, the first page contains the first 2000 entities and contains a next link at the end of the response. The next link contains the URL to fetch the next page of results. Next link URLs should be passed back without modification. You can modify the maximum number of entities returned in a page by setting the OData Page Size data source parameter as described in Configuring data sources for OData Version 2 connectivity on page 623.

- Client-side pagination is controlled by both the client and the Hybrid Data Pipeline OData service. Requests can specify a particular page size with the $top query parameter and can navigate through the pages by specifying different values for the $skip query parameter. The Top Mode setting allows the Hybrid Data Pipeline service to optimize queries in certain situations. You can set the Top Mode in the data source definition or use the ddcloud-top-mode header in requests to inform the service of how the client uses $top. See Configuring data sources for OData Version 2 connectivity on page 623 and Top Mode on page 844 for more information.

For example, the following URL requests Employees entities in pages of 100.

https://<myserver>:<port>/api/odata/OracleOPTest/EMPLOYEES?$top=100&amp;skip=0&amp;format=json
To fetch the next page, increment the $skip parameter by the page size.

https://<myserver>:<port>/api/odata/OracleOPTest/EMPLOYEES?$top=100&$skip=100&$format=json

The client can request any page size it needs. However, the Hybrid Data Pipeline connectivity service might return fewer entities than were requested. In this case, the response will contain a next link, as with server-side paging. The client should use the next link(s) to get all of the results before requesting the next page.

Custom query parameters

Hybrid Data Pipeline OData service provides the following custom query parameters.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>timezone</td>
<td>A Java timezone id string. If the client timezone differs from that of the Hybrid Data Pipeline service, specifying the timezone might be necessary to correctly process DateTime values. The timezone can also be specified as header. See OData Headers for more information.</td>
<td>When not specified in the URL or as a header, defaults to GMT.</td>
</tr>
<tr>
<td>ddsearch</td>
<td>Use in queries with a string to search columns for which search is enabled in the schema map, in contrast with $filter, which searches all exposed columns. Do not use ddsearch and $filter in the same request. See Searching text-based columns with OData Version 2 on page 849 for more information.</td>
<td>Not applicable.</td>
</tr>
</tbody>
</table>

Searching text-based columns with OData Version 2

Different data store types support different levels of indexing and searching. Indexing increases the efficiency of searches in tables with many records. Querying to find particular values can be expensive when the search must span many columns and many records. To improve performance, you can restrict searches to particular text-based columns using the Hybrid Data Pipeline proprietary query parameter, ddSearch. To search across all columns in the schema, even those not enabled in the schema map for searching, you can use OData $filter. But, you cannot combine ddSearch and $filter in the same request.

This release supports use of ddSearch for all data store types, and full-text search taking advantages of indexes in the following data source types:

- DB2 on Linux, UNIX, and Windows — Each column to be searched must have a separate full text index, the full text services must be running, and the database must be enabled for full text. See the DB2 documentation for more information.
- Oracle — Each column to be searched must have a separate full text index, the full text services must be running, and the database must be enabled for full text. See the Oracle documentation for more information.
- Microsoft SQL Server — Each column to be searched must have a separate full text index and the full text index engine must be running. See the Microsoft documentation for more information.
To use text search:

- For data stores that support full-text search, make sure that the underlying data store is indexed and is up to date with the current schema.

- For Salesforce data stores that access external objects, follow the steps described in Configuring Salesforce external objects for search optimization on page 850.

- Enable search for the indexed columns in the Hybrid Data Pipeline data source schema map, as described in Configuring data sources for OData Version 2 connectivity on page 623 and selecting Full Text as the search type.

- Use the ddsearch parameter with a search string, as described below.

Hybrid Data Pipeline treats multiple terms by using a logical `and`. For example, a search for Sales & Marketing returns records that contain both the word Sales and the word Marketing, the ampersand is ignored. The case-sensitivity of the search string depends on the underlying data source.

The ddsearch parameter will either return an empty response or an error in the following circumstances:

- If the schema map does not specify the table as searchable.

- If the table does not contain searchable fields.

- If searching is not enabled in the backend data store.

- For Salesforce, if you have not enabled use of ddSearch as a custom query parameter.

The following example returns a list of records containing the string "TX" from an ACCOUNT table:

https://<myserver>:<port>/api/odata/DDCdemo/ACCOUNTS?ddsearch=TX

where `<myserver>` is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

Configuring Salesforce external objects for search optimization

Hybrid Data Pipeline provides the ability to configure which tables and columns are included in a search to optimize performance and avoid overloading database resources. If you use Salesforce to access external objects, and want to take advantage of Hybrid Data Pipeline's optimization, you must configure the Salesforce external data source to accept the Hybrid Data Pipeline `ddsearch` parameter. This can improve the performance of the OData queries generated by Salesforce to search your external objects.

To do this for an existing external data source, log into your Salesforce account and follow these steps:

- Navigate to the External Data Source Edit screen.

- Make sure that Include in Salesforce Searches is enabled.

- In the Custom Query Option for Salesforce Search field, enter ddsearch as shown below:

- Save your changes.

Note: Navigation to the External Data Source screen differs depending on the type of Salesforce account you have. See your Salesforce documentation for more information.
Fetching records and collections with OData version 2

As shown in the following table, use the plural entity name with the GET method to fetch metadata, a single entity, an entity’s property, or a collection of entities. When using a data source group, prepend the entity name with the appropriate data source prefix. See URI conventions for addressing resources, entities, and related entities in Section 3 of the OData specification.

<table>
<thead>
<tr>
<th>To fetch:</th>
<th>Method:</th>
<th>URI:</th>
</tr>
</thead>
<tbody>
<tr>
<td>A single record</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_singlar_name&gt;('primary_key_value')</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td><a href="https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS('1">https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS('1</a>')</td>
</tr>
<tr>
<td>The value of a single field from a single record</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_singlar_name&gt;('primary_key_value')/&lt;column_name&gt;/$value</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td><a href="https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS('1')/NAME/$value">https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS('1')/NAME/$value</a></td>
</tr>
<tr>
<td>A collection of records*</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_plural_name&gt;</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td><a href="https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS">https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS</a></td>
</tr>
<tr>
<td>A count of the records in a collection</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_plural_name&gt;$count</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
<td><a href="https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS/$count">https://myserver:8080/api/odata/MySFDataSource/ACCOUNTS/$count</a></td>
</tr>
</tbody>
</table>

*A single request can only fetch one collection.

Creating, editing, and deleting records with OData Version 2

Create records using the POST method. Update and delete records using the POST method with the custom header, X-HTTP-Method, with a value of MERGE or DELETE. A request should include:

- Your Hybrid Data Pipeline account credentials.
- If the backend data source credentials are not stored in the Data Source definition, the ddcloud-datasource-user and the ddcloud-datasource-password headers.
- The resource URL appropriate for the operation:
  - To create a record, include the plural entity name and supply property values in the body.
  - To update a record, include the plural entity name and the primary key value.
  - To delete a record, include the plural entity name and the primary key value.

To create or update, supply property values in either Atom Pub or JSON format. Use the Content-Type header to specify the format as one of the following:

- `application/atom+xml`
Create example

When supplying property values, include required columns (except for those with default values or set automatically by the data store). The following screen shows a POST request in Postman to create an ACCOUNT entity in a Salesforce data store. To formulate the request:

- The header **Content-Type** has the value `application/atom+xml`.
- The URL includes:
  - The service root, `<myserver>:<port>/api/odata`.
  - The **Data Source** definition name, `sfds`.
  - The plural entity name, `ACCOUNTS`.

- The body includes:
  - The value of the `entry` element and structure of the `m:property` element were copied from the response of a GET request that fetched a single account record.
  - No value was supplied for `ROWID`, the primary key, because Salesforce generates the value automatically.
The following lines in the response show that the new record was successfully created:

```
<entry>
  <title type="text"/>
  <updated>2016-01-20T15:32:48Z</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="ACCOUNT" href="ACCOUNTS('001i0000001mDL2qAAG')">'/self_ref_ACCOUNT'/
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/realted/self_ref_ACCOUNT" type="application/atom+xml;type=entry" title="self_ref_ACCOUNT" href="ACCOUNTS('001i0000001mDL2qAAG')/self_ref_ACCOUNT" />
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/realted/ACCOUNTS" type="application/atom+xml;type=feed" title="ACCOUNTS' href="ACCOUNTS('001i0000001mDL2qAAG')/ACCOUNTS'/
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/realted/ACCOUNTS(‘001i0000001mDL2qAAG’)/LEADS’/>
  <category term="DDCdemo.ACCOUNT" scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme'/
  <content type="application/xml'/
    <d:ROWID>001i0000001mDL2qAAG</d:ROWID>
    <d:SYS_NAME>Rainbow Ice Pops</d:SYS_NAME>
    <d:BILLINGSTREET>3466 Brody Un</d:BILLINGSTREET>
    <d:BILLINGCITY>Austin</d:BILLINGCITY>
    <d:BILLINGSTATE-TX</d:BILLINGSTATE>
    <d:ACCOUNTNUMBER m:nullable="true"/>
  </m:properties>
</entry>
```

For more details on creating records and an example in JSON format, see HTTP POST (create) on page 860.

**Delete example**

To delete a record, use HTTP DELETE or the POST request with the custom X-HTTP-Method header value of DELETE. Supply the primary key of the record to delete. The following screen shows a request in Postman to delete an account name from a Salesforce data store.

To formulate the request:

- The **Content-Type** header value is application/atom+xml.
- The custom header **X-HTTP-Method** value is DELETE.
- The resource URL includes:
  - The **service root**, `<myserver>:<port>/api/odata`.
  - The **Data Source** definition name, sfds.
  - The plural entity name, ACCOUNTS followed by the primary key (partially cut off in the screen shot).
- The body of the request is empty.
The following screen shows the result of executing the request. The **Status** of **204 No Content** indicates that the record was successfully deleted.

### Update example

To update a record, use a **POST** request with the custom **X-HTTP-Method** header. Supply the primary key in the resource URL and the property value(s) for the column(s) to update in the body. The following screen shows a request in Postman to update an account name from **Hot Diggity Dog** to **Hot Diggity Dogs** in a Salesforce data store. To formulate the request:

- The **Content-Type** header value is **application/atom+xml**.
- The custom header **X-HTTP-Method** value is **MERGE**.
- The URL includes:
  - The service root, `<myserver>:<port>/api/odata`.
  - The **Data Source** definition name, `sfds`.
  - The plural entity name, `ACCOUNTS` followed by the primary key, `001i000001mDKrJAAW` (which is cut off in the screen shot).

- The body includes:
  - The value of the `entry` element and structure of the `m:property` element were copied from the response of a GET request that fetched a single account record.
  - A value of **Hot Diggity Dogs** for the **SYS_NAME** property.
The **Status value** 204 No Content shown in the screen above indicates that the name was successfully updated. A fetch of the record confirms the update to Hot Diggity Dogs as shown below:

```
<entry xmlns="http://www.w3.org/2005/Atom"
>
  <m:properties>
    <d:ROWID>mDE3JjAw</d:ROWID>
    <d:SYS_NAME>Hot Diggity Dogs</d:SYS_NAME>
    <d:BILLINGSTREET>2335 Wilson Lane</d:BILLINGSTREET>
    <d:BILLINGCITY>Austin</d:BILLINGCITY>
    <d:BILLINGSTATE>TX</d:BILLINGSTATE>
    <d:ACCOUNTNUMBER m:null="true"/>
  </m:properties>
</entry>
```

For more information on updating, see HTTP POST and MERGE (update) on page 859

**Navigating relationships with OData Version 2**

Most data source types supported by Hybrid Data Pipeline use relationships to define associations between tables or objects. In a relational data source, foreign key columns reference the primary key column of the related table. When you configure a schema map for a data source that contains relationships, Hybrid Data Pipeline maps them as OData relationships. The OData model (returned via $metadata) identifies these as Navigation Properties. OData provides the following ways to access related entities:

- **Resource Path navigation** — fetch all related records or a specific record or property of that record.
- **$Links** — fetch all records for an entity and embed all related records in the response.
- **$expand** — return links to the related records for a specific entity.
Hybrid Data Pipeline supports all three ways of navigating relationships. The topics in this section use an example of customers and orders with the following model:

```
Customer ----> Order ----> OrderItem
    |                ---> Contact
```

**Resource path navigation**

Resource path navigation allows a query to reference a related entity from a parent or child entity. For example, with the following table structure, a customer's orders can be referenced from a Customer record, as shown below.

```
Customer ----> Order ----> OrderItem
    |                ---> Contact
```

**List the orders for a particular customer**

https://<myserver>:<port>/api/odata/OracleDS/Customers('3')/Orders

**List the order items for a particular order for customer 3**

https://<myserver>:<port>/api/odata/OracleDS/Customers('3')/Orders('5')/OrderItems

**Access a particular order item**

https://<myserver>:<port>/api/odata/OracleDS/Customers('3')/Orders('5')/OrderItems('6')

**Access a particular property**

https://<myserver>:<port>/api/odata/OracleDS/Customers('3')/Name

https://<myserver>:<port>/api/odata/OracleDS/Customers('3')/Orders('5')/OrderItems('6')/ItemName

**$links construct**

The examples in this topic use the following table structure:

```
Customer ----> Order ----> OrderItem
    |                ---> Contact
```

$links navigation is similar to Resource path navigation except that instead of returning the data for the referenced resource, a link to the referenced resource is returned.

For example, a query that lists the orders of a particular customer could be written as:

https://<myserver>:<port>/api/odata/SQLServerDS/Customers('3')/$links/Orders

where `<myserver>` is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed. This returns links to the orders that belong to customer 3.
$expand query parameter

The examples in this topic use the following table structure:

```
Customer ----> Order ----> OrderItem
   |        ---> Contact
```

The $expand system query parameter allows the related information to be embedded in the response of the parent or child entity. For example, you can obtain a list of customers with a list of all of their orders by issuing the query:

```
https://<myserver>:<port>/api/odata/OracleDS/Customers?$expand=Orders
```

Each customer entity in the response contains the list of order entities belonging to that customer embedded in the customer entity. Multiple tables can be expanded. The following query returns the list of customer entities; embedded in each customer entity is the list of their orders and the list of contacts for that customer.

```
https://<myserver>:<port>/api/odata/OracleDS/Customers?$expand=Orders, Contacts
```

Hybrid Data Pipeline currently only allows expanding to one level deep. For example, the following multi-level query, which attempts to expand orders and order items for a customer, is not currently supported:

```
https://<myserver>:<port>/api/odata/OracleDS/Customers?$expand=Orders/OrderItems
```

Method Reference for OData Version 2

The Hybrid Data Pipeline OData service interface supports GET, POST, POST/MERGE and POST/DELETE HTTP methods. Each operation acts on the resource specified in the URL.

The POST request to create or update an entity should include a Content-Type header specifying the format of the request payload. The Hybrid Data Pipeline OData API recognizes the following content types:

- application/atom+xml
- application/atom+xml;charset=UTF-8
- application/json
- application/json;charset=UTF-8

If the Content-Type header is not supplied, Hybrid Data Pipeline interprets the body as the Atom Pub format encoded using the character set ISO-8859-1 character set.

Supported OData API Operations

The following table shows the operations that can be performed and their associated URLs. Refer to the specified section for detailed descriptions for these operations. Query the data source name to get a list of the valid entities.

In this table, <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.
<table>
<thead>
<tr>
<th>Purpose</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fetch Data from an OData Service</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/odata/&lt;data-source-name&gt;&lt;entity-plural-name&gt;</td>
</tr>
<tr>
<td>Create an Entity</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/odata/&lt;data-source-name&gt;&lt;entity-plural-name&gt;</td>
</tr>
<tr>
<td>Delete an Entity</td>
<td>DELETE Or POST X-HTTP-Method:DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/odata/&lt;data-source-name&gt;&lt;entity-plural-name&gt; {'primary-key'}</td>
</tr>
</tbody>
</table>

### HTTP GET

**Purpose**

Fetch an entity, collection of entities, or a property of an entity. The authenticated user must be the owner of the data source requested. If the authenticated user is not the owner of the data source, a "data source not found" error is returned.

**URL**

https://<myserver>:<port>/api/odata/<resource path>

where

where `<myserver>` is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

`<resource path>` is the address of an entity, entity collection, or a property of an entity. See Service URI and resource path on page 845 for more information on addressing entities.

**Method**

GET

**Response**

A JSON or Atom Pub representation of the entity, entity collection, or entity property specified in the URL.

**Authentication**

Basic Authentication using the Hybrid Data Pipeline account user ID and password.
Authorization
Any active Hybrid Data Pipeline user. The authenticated user must use same credentials used to create the data source definition.

See also
Creating an Entity on page 860

HTTP DELETE or POST and DELETE

Purpose
HTTP DELETE deletes a specified entity. Alternatively, you can use HTTP POST and specify DELETE as the value of the X-HTTP-Method header. The body of the request must be empty and the URL should not contain parameters.

URL
https://<myserver>:<port>/api/odata/<entity collection>/<entity instance>
where <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

Method
DELETE | POST with a X-HTTP-Method header value of DELETE.

ResponseStatus
If the entity is successfully deleted, the OData service returns a status of 204 No Content.

Authentication
Basic Authentication using Login ID and Password.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must use same credentials used to create the data source definition.

Sample Requests
DELETE https://myserver:8080/api/odata/Customer(123)

POST https://service.datadirectcloud.com/api/odata/Customer(123)
X-HTTP-Method: DELETE

HTTP POST and MERGE (update)

Purpose
Update an entity using the custom X-HTTP-Method header with a value of MERGE. The body of the request should contain an entity description of the properties of the entity to be changed.
Note: Hybrid Data Pipeline supports neither HTTP UPDATE nor OData PUT semantics.

URL
https://<myserver>:<port>/api/odata/<entity collection>/<entity instance>
where <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

Method
POST

Syntax
The request uses the following format:

POST <base>/Customers(123)
accept: application/<content-type>[,<content-type>]
X-HTTP-Method: MERGE

Response Status
If the entity is successfully updated, the OData service returns a 204 No Content status.

Restrictions
You cannot update a property that is part of the primary key; if you supply a value, Hybrid Data Pipeline will ignore it. If a property in the entity description does not correspond to a property in the entity, then an error with a 400 Bad Request status is returned.

An HTTP request with the method set to MERGE is not supported and will return a 405 Method Not Supported response status.

Authentication
Basic Authentication using Login ID and Password. The authenticated user must use same credentials used to create the data source definition.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the data source.

HTTP POST (create)

Purpose
Create an entity in an existing entity collection — a table or object in the underlying data store. The body of the POST request describes the entity to be created and can be specified in the JSON or the Atom Pub (XML) OData format. Use the Content-Type header to specify the format.
Entity descriptions include the following:

- Values for all required properties, which include those that map to an updateable column in the data store that is defined as NOT NULL, that does not have a default value, and is not automatically generated by the data source.

- Optionally, include values for property values that cannot be updated. However, in this release, Hybrid Data Pipeline ignores these values.

- Optionally, specify values for navigation properties to create a relationship with other records.

**URL**

https://<myserver>:<port>/api/odata/<data source name><entity collection path>

where <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

**Method**

POST

**Response**

The body of the response contains the value of the new entity in the same format in which the entity definition was provided in the request. The entity value returned includes the correct values for any computed or auto-generated properties, and the Location header. The value of the Location header is the URL of the entity inserted. For example, the location header for the entity created in the preceding example may have the value.

https://myserver:8080/api/odata/myoracle/Products(10)

**Response Status**

If the entity is created successfully, the OData service returns a 201 Created status. The body of the response contains the value of the new entity in the same format as the entity definition provided in the request. The entity value returned includes the correct values for any computed or auto-generated properties, as well as the Location header, which contains the URL of the entity created.

If the value for a required property is omitted from the entity description, the OData service returns a 400 Bad Request response. The message provides an indication of which required property was not specified.

**Authentication**

Basic Authentication using the Hybrid Data Pipeline user ID and password. The credentials used for the request must be the same credentials used to create the data source definition.

**Authorization**

Any active Hybrid Data Pipeline user. The authenticated user must use same credentials used to create the data source definition.

**Sample Request Payload**

The following example uses the JSON format to create a new Product entity in an Oracle data source.

```
POST https://myserver:8080/api/odata/myoracle/Products
{
   "ID" : 10, "Name" : "Hosta",
   "Description" : "With new features",
   "ReleaseDate" : "\Date(1436342315266)\",
   "Rating" : 1,
```
"Price" : "1.23"
}
Querying with OData Version 4

For details, see the following topics:

• Getting started with OData Version 4
• Supported functionality for OData Version 4
• Understanding and configuring a schema map for OData Version 4
• Structure requests for OData Version 4
• Formulating queries with OData Version 4
• Method Reference for OData Version 4

Getting started with OData Version 4

This section describes using Hybrid Data Pipeline to query data with OData Version 4. Hybrid Data Pipeline also supports OData Version 2. For information on querying with OData Version 2, see Getting started with OData Version 2 on page 827.

The Open Data Protocol (OData) provides a standard for exposing resources using Uniform Resource Identifiers (URIs) and an API for querying the resources with simple HTTP messages. Hybrid Data Pipeline OData services support OData requests for a variety of data stores. Since OData is REST-based, and does not require any locally-installed software, the Hybrid Data Pipeline OData API provides quick and easy data access for mobile apps and desktop applications.
The OData API is based on an object model instead of the tabular representation used by many data stores. To translate OData requests, Hybrid Data Pipeline requires a schema map. As part of a data source definition, you use the Configure Schema editor to select the tables (or objects), columns (or attributes) and functions to access with OData. Hybrid Data Pipeline generates a JSON schema map that exposes your selections as entities and their properties.

**Using OData**

To access a data store using OData requires both Hybrid Data Pipeline configuration and implementation on the client-side.

1. While logged into Hybrid Data Pipeline, create or edit a data source definition.
2. In the data source definition, enable OData access by Configuring data sources for OData Version 4 connectivity on page 627.
3. In the client, create requests to the OData-enabled data source, as demonstrated in Testing data source configurations (OData Version 4) on page 872 and described in more detail in Formulating queries with OData Version 4 on page 892.

**Configuring data sources for OData Version 4 connectivity**

Hybrid Data Pipeline supports OData Version 2 and Version 4 connectivity for all supported data stores. You can configure a data source on any data store for OData connectivity either during the process of creating the data source or after the data source has been created.

The following steps describe how to configure a data source for OData Version 4 connectivity.

1. From the Web UI, navigate to the Data Sources view by clicking the data sources icon.
   - **Option 1.** If creating a new data source, click New Data Source, choose the data store, enter the required information on the General tab, and click TEST to confirm connectivity to the backend data store. (See Creating data sources with the Web UI on page 224 for details.)
   - **Option 2.** If enabling OData on an existing data source, select the data source you wish to modify.
2. Select the OData tab.
3. For **OData Version**, select **Version 4**.

4. Open the Configure Schema editor by clicking **Configure** to the right of the **Schema Map** field.

5. Select a schema from the **Select Schema** dropdown.

   **Note**: By default, Hybrid Data Pipeline exposes all schemas on any backend data stores that support multiple schemas. The **Metadata Exposed Schemas** option on the **Advanced** tab for any such data store can be used to limit exposed schemas to a single schema. If a schema is selected for the **Metadata Exposed Schemas** option, it will be the only schema available on the Configure Schema editor's **Select Schema** dropdown.

6. From the **Tables and Columns** tab, select and define the tables and columns you want to expose to OData client applications.
• To add all tables, click **Add All Tables** on the **Tables** panel.

• To add individual tables, select a table on the **Tables** panel and click **Add To Map** in the **Settings** panel to the right.

• To remove a table that was previously added, select the table and click **Remove From Map** in the **Settings** panel.

• To specify singular and plural alias names for a table, select the table, enter the table alias for the entity type name in the **Singular Name** field, enter the table alias for the entity collection name in the **Plural Name** field, and click **Add To Map**.

**Note:** The singular alias name specified is used as the entity type name, while the plural alias name will be used as the entity collection name. When alias names are not specified, the mapping of entity names will be dictated by the **Entity Name Mode** setting in the **OData Settings** tab, as described in Step 9.

• To specify a column as a primary key, select the column from the **Columns** panel and set the **Is Primary Key** switch from **OFF** to **ON**.

**Note:** The **Configure Schema** editor indicates that a primary key exists for a table with a skeleton key icon. A primary key assigned in the backend data store cannot be changed. If a primary key has not been discovered for a table you wish to map, one or more columns must be specified as a primary key.

• To remove a column from the OData schema map, select the column from the **Columns** panel and click **Remove From Map** in the **Settings** panel.

**Note:** When a table is added, all columns in the table are exposed in the OData schema map by default. You can modify the columns exposed by removing (or excluding) them from the schema map.

7. From the **Tables and Columns** tab, select the columns you want to view or modify.
• To specify an alias name for a column, select the column and enter an alias in the **Alias Name** field. If specified, the alias name will be used as the OData name for the column. If not specified, the name of the column will be used as the OData name.

• To specify a column as a primary key, set the **Is Primary Key** switch from **OFF** to **ON**.

**Note:** The Configure Schema editor indicates that a primary key exists for a table with a skeleton key icon. A primary key assigned in the backend data store cannot be changed. If a primary key has not been discovered for a table you wish to map, one or more columns must be specified as a primary key.

• Open **Advanced Settings** to review and modify column metadata. The **Advanced Settings** allow you to modify column metadata returned by the underlying JDBC driver. This is especially useful when the JDBC driver returns incorrect metadata. The **Driver Value** of each setting indicates the value that is returned by the driver. You can specify settings related to the following properties:

  • **Data Type:** Indicates the data type for the column. If you wish to use the **Actual Value**, you can leave the Data Type as **Default**. If you wish to override the data type specified, you can choose an alternate data type from the dropdown list.

  **Note:** Depending on the data types selected, some of the Advanced settings options will be enabled or disabled. For example, **Scale** is enabled for the decimal datatype, and not for the integer datatype.

  • **Column Size or Precision:** Indicates the maximum precision or maximum length of the column.

  • **Scale:** Indicates the maximum scale of the column.
• **Is Nullable**: Indicates whether the column can have a null value. Normally drivers report this correctly. Some drivers may report a column as not nullable while null values exist in the column. In such a scenario, the **is Nullable** could be set to true to correct this issue. Note that there could be implications on the create entity behavior by changing this setting.

• **Is Auto Increment**: Indicates whether the column is a uniquely generated column. Setting this to true will indicate to the service that it should ignore incoming values for this column during the create, update, and patch entity operations.

• **Is Generated**: Indicates whether the column is a generated value. If the column is generated, then the OData code will ignore incoming values for this column during the create, update, and patch entity operations.

8. Take the following steps to enable text search for individual tables and text-based columns using the `$search` system query option.
   a) Select a table from the **Tables** panel.
   b) Specify a search option from the **Search Options** dropdown. Then click **Add To Map**.
      - **Full Text** is only available for data store types that support indexing and full text search.
      - **Substring** enables searches for the string anywhere in the search-enabled fields.
      - **Begins** restricts the search to the text at the beginning of a field.
   c) If you selected **Full Text** in Step b, you should select an index type for all text-based columns. Select the column from the **Columns** panel, and specify an index type from the **Index Type** dropdown in the **Settings** panel. Then click **Add To Map**.
      The index type is the type of index supported by the backend data store. **TEXT** is the only valid value for the DB2 and SQL Server data stores. **CONTEXT** and **CTXCAT** are the valid values for the Oracle data store. If **Full Text** has been selected but the data store index has not been properly configured, queries using `$search` will return errors.
   d) If you selected **Substring** or **Begins** in Step b, you should select which text-based columns can be searched. Select the column from the **Columns** panel, and set the **Is Searchable** switch to **ON**. Then click **Add To Map**.

9. Take the following steps to expose stored functions. (Note that stored functions are supported only for DB2, SQL Server, and Oracle data stores. See **Stored functions support** on page 880 for details on further restrictions).
a) Select the **Functions** tab.

b) Select the function you want to expose from the **Functions** panel.

c) If desired, specify an alias name for the stored function.

d) If desired, specify an import alias name for a function import that corresponds to the function.

e) Specify whether the OData type is a function or an action on the **OData Type** dropdown.

f) Click **Add To Map**.

10. Specify general settings on the **OData Settings** tab. Then click **Add To Map** to apply settings.

- From the **Entity Name Mode** dropdown, specify the algorithm used to map table names to entity collection names or entity type names. Entity collection names are usually plural, while entity type names are usually singular.
When **guess** (default) is selected, one of the following algorithms is applied based on an evaluation of the table name.

- If the table name ends with a numeric digit, the table name is used as the entity collection name and a suffix is appended to the table name for the entity type name. The suffix used can be specified in the **Singular Suffix** field.
- If the table name does not end with a digit and appears to be **singular**, the table name is used as the entity collection name and singularized for the entity type name.
- If the table name does not end with a digit and appears to be **plural**, the table name is used as the entity type name and pluralized for the entity collection name.

When **singularize** is selected, the table name is used as the entity collection name. The table name is then singularized for the entity type name.

When **pluralize** is selected, the table name is used as the entity type name. The table name is then pluralized for the entity collection name.

When **suffix** is selected, the table name is used as the entity collection name. For the entity type name, a suffix is appended to the table name. The suffix used can be specified in the **Singular Suffix** field.

With the **Time As String** switch, specify how the JDBC type Time should be mapped.

- If set to **OFF** (default), Time is mapped to the OData type TimeOfDay.
- If set to **ON**, Time is mapped as String.

In the **Singular Suffix** field, enter the suffix that will be appended to an entity type name when the **Entity Name Mode** has been set to either **guess** or **suffix**.

With the **Unbound Number as Double** switch, specify whether decimal columns and parameters with no precision or scale should be automatically mapped as Double.

- If set to **OFF** (default), decimal columns and parameters with no precision or scale are not automatically mapped as Double.
- If set to **ON**, decimal columns and parameters with no precision or scale are automatically mapped as Double.

11. Click the **Review Schema Map** tab to review the OData schema map in JSON format.
12. Click **Save Map** to save your configuration of the OData schema map.

13. Set OData options to the desired values.

- **Data Source Caching** controls caching of results. A value of 0 results in a stateless session, requiring Hybrid Data Pipeline to access the data store for each request. A value of 1, the default, allows Hybrid Data Pipeline to cache results, improving performance for repeated requests for the same entity. Hybrid Data Pipeline clears the cache after ten minutes of inactivity or at the end of a session.

- **Page Size** controls the number of results returned in one response. By default, the value in this field is 0 which causes Hybrid Data Pipeline to return up to 2,000 top-level entities per response. If the response contains more than 2,000 entities, the first 2,000 entities are returned and the end of the response contains a link that the OData client can use to fetch the next set. You can set the page size by using values from 1 to 10,000. Client requests can also specify the size of results with query parameters.

- **Refresh Result** determines whether Hybrid Data Pipeline returns results from the cache (for entities in the cache) or queries the data source again. A value of 1, the default, allows Hybrid Data Pipeline to satisfy requests from cached results. A value of 0 forces queries to the backend data store. If caching is not enabled, this parameter has no effect.

- **Inline Count Mode** controls how Hybrid Data Pipeline handles requests that include the $inlinecount parameter with a value of allpages. The response includes the total number of entities that satisfy the query. A value of 0 causes Hybrid Data Pipeline to skip counting. A value of 1 causes Hybrid Data Pipeline to run a separate query to get the count before the query that returns the entities. This can result in the first page of results being returned faster for large result sets for some data store types. A value of 2, the default, causes Hybrid Data Pipeline to fetch all results and calculate the total number before returning the first page of results to the client.

- **Top Mode** allows Hybrid Data Pipeline to better handle requests that include the $top parameter. A value of 0, the default, indicates that clients using $top to limit result set size will rarely attempt to get additional entities using the $skip parameter. A value of 1 indicates that clients generally use $top and $skip together to paginate results.

- **OData Read Only** controls read/write access. For a new data source definition, this option is not selected by default. For a data source definition where OData was enabled before this option was available, it will be checked by default. Remove the check mark to enable write access.

14. Click **Update** to save your work.
What to do next:

Test your OData-enabled data source as described in Testing data source configurations (OData Version 4) on page 872.

After you create an OData-enabled data source, you can view the status of the schema map generation on the Data Sources screen. The icon besides the OData-enabled data source indicates the status of the schema map generation. The following table provides details of the icons.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image" /></td>
<td>The synchronization of the schema map is in progress. The number denotes the percentage of synchronization completed.</td>
</tr>
<tr>
<td><img src="image2" alt="Image" /></td>
<td>The schema map was synchronized successfully.</td>
</tr>
<tr>
<td><img src="image3" alt="Image" /></td>
<td>The schema map was synchronized successfully, but there are some table/column warnings. Hybrid Data Pipeline allows users to know the details of the tables/columns and/or functions that were dropped while generating the OData Model for a given schema map of a Data Source. The number of warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the Schema API on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td><img src="image4" alt="Image" /></td>
<td>Errors occurred while synchronizing the schema map. You must address the errors and synchronize the schema map again. Hybrid Data Pipeline allows users to know the details of the tables and/or columns that were dropped while generating the OData Model for a given schema map of a Data Source. The number of errors/warnings shown is limited to 100. If there are more than 100 errors/warnings, you can use the Schema API on page 1412 to retrieve table and column warnings.</td>
</tr>
<tr>
<td><img src="image5" alt="Image" /></td>
<td>You must synchronize the schema map again.</td>
</tr>
</tbody>
</table>

Testing data source configurations (OData Version 4)

You can quickly test the configuration from the Hybrid Data Pipeline dashboard or by using a REST client.

- Testing data source configurations from the Hybrid Data Pipeline dashboard on page 872.
- Testing data source configurations using a REST client on page 873
- Testing OData functions using a REST client on page 874

Testing data source configurations from the Hybrid Data Pipeline dashboard

Take the following steps to test whether your data source definition and schema map are configured correctly.
1. In the left navigation pane, select Data Sources to open your list of data sources.

2. Select the OData-enabled data source definition, and click the OData URI icon at the end of the row.

3. Enter your Hybrid Data Pipeline credentials.

   The browser returns an XML document listing the entities in the schema.

   ![XML document with entities](image)

   **Testing data source configurations using a REST client**

   Take the following steps to test a data source configuration using a REST client. In this example, Postman is used as the REST client.

   1. Using the controls exposed by the REST client, select basic authorization and enter your Hybrid Data Pipeline credentials.

   2. If credentials for your data store are not saved in the data source definition, pass them as values for ddcloud-datasource-user and ddcloud-datasource-password headers.

   3. From the OData tab of the data source you are testing, copy the OData Access URI. Then paste the URI in the URL field of the REST client.

   4. Execute a GET on the data source endpoint. For example:

      ```
      GET https://service.myserver.com/api/odata4/db2ds
      ```

   The response payload returns a list of entities exposed by the OData schema map.
Testing OData functions using a REST client

Take the following steps to test OData function invocation using a REST client. Again, Postman is used as the REST client.

1. Using the controls exposed by the REST client, select basic authorization and enter your Hybrid Data Pipeline credentials.

2. If credentials for your data store are not saved in the data source definition, pass them as values for `ddcloud-datasource-user` and `ddcloud-datasource-password` headers.

3. From the OData tab of the data source you are testing, copy the OData Access URI. Then paste the URI in the URL field of the REST client. Append the URI with the `$metadata` endpoint.

4. Execute a GET on the `$metadata` endpoint. For example:

   ```
   GET https://service.myserver.com/api/odata4/sample_datasource/$metadata
   ```

   The response payload returns the OData schema map in XML format. The schema map includes any functions exposed in the OData model.

   ![XML Schema Map](image)

   The function import name can be used to invoke the function independently.

   ```
   GET https://service.myserver.com/api/odata4/sample_datasource/OData_sample_function?INT64=10
   ```

   ![Function Invocation](image)

   Table names can be used to return data stored in tables.
The function name can be used to invoke the function as part of a $filter query.

Requesting service metadata and the service document

Metadata for your OData service can be fetched by requesting the service document or service metadata using a GET request.

Service Document

The service document returns a list of all the available entities in a schema in the request payload. To fetch the service document, issue a GET request for the data source's service root:

```xml
<server>:<port>/api/odata4/<hdp_data_source>
```

For example:

https://MyServer:8443/api/odata4/myds/

Service Metadata

Fetching service metadata returns a description of the data model for the service, including the names, properties, data types, and relationships for all entities in the schema. To fetch service metadata, issue a GET request for the data source's service root with /$metadata appended to the path:

```xml
<server>:<port>/api/odata4/<hdp_data_source>/$metadata
```

For example:

https://MyServer:8443/api/odata4/myds/$metadata
You can use the odata.metadata parameter in the Accept header to determine the level of control information returned for $metadata requests. For example:

```plaintext
GET https://MyServer:8443/api/odata4/myds/$metadata
OData-Version: 4.0
Content-Type: application/json;odata.metadata=full
Accept: application/json
```

The level of information returned can be set to full, minimal, or none, depending on the needs of your application. full provides the most annotations, but at greater expense to the wire, while none returns the fewest at the least expense. The following table provides a list of the required annotations returned by level.

<table>
<thead>
<tr>
<th>Level</th>
<th>Annotations Returned</th>
</tr>
</thead>
</table>
| full           | • odata.context
                | • odata.count
                | • odata.nextLink
                | • odata.id
                | • odata.type |
| minimal (default) | • odata.context
                | • odata.count
                | • odata.nextLink |
| none           | • odata.count
                | • odata.nextLink |

**Supported functionality for OData Version 4**

Hybrid Data Pipeline supports the OData Version 4.0 and Version 2.0 specifications.

Data sources and data source groups support using a single supported version of the specification at a time. The version used by a data source is determined by the setting of the OData Version parameter on the OData tab. The OData version of a data source group must match the OData version of its member data sources.

This section describes using Hybrid Data Pipeline with OData Version 4. For information on using Hybrid Data Pipeline with OData Version 2.0, see Getting started with OData Version 2 on page 827
Supported OData operations and data types

Supported OData API Operations

The following table shows the operations that can be performed and their associated URLs. Query the data source name to get a list of the valid entities.

In the URL examples in this table,

- `<myserver>` is the DNS name or the IP address of the machine on which Hybrid Data Pipeline is installed.
- `<myds>` is the name of your Hybrid Data Pipeline data source.
- `<plural-name>` is the name you designate in your schema map for entity plurals. In the schema map, Hybrid Data Pipeline pluralizes the table name automatically. You use the plural entity name in OData requests.
- `pkey` is the primary key.
<table>
<thead>
<tr>
<th>Purpose</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fetch Data from an OData Service</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:8443/api/odata4/&lt;myds&gt;/&lt;plural-name&gt;</td>
</tr>
<tr>
<td>Create an Entity</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:8443/api/odata4/&lt;myds&gt;/&lt;plural-name&gt;</td>
</tr>
<tr>
<td>Update an Entity</td>
<td>PATCH</td>
<td>https://&lt;myserver&gt;:8443/api/odata4/&lt;myds&gt;/&lt;plural-name&gt;('pkey')</td>
</tr>
<tr>
<td>Or</td>
<td>POST</td>
<td>X-HTTP-Method:PATCH</td>
</tr>
<tr>
<td>Delete an Entity</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:8443/api/odata4/&lt;myds&gt;/&lt;plural-name&gt;('pkey')</td>
</tr>
<tr>
<td>Or</td>
<td>POST</td>
<td>X-HTTP-Method:DELETE</td>
</tr>
</tbody>
</table>

**Entity Data Model (EDM) types for OData Version 4**
To support communication between an OData client and a backend data store, Hybrid Data Pipeline uses a schema map to convert data to the appropriate type for the receiver. You configure the schema map in Hybrid Data Pipeline where it is generated as a JSON string with the following OData Entity Data Model (EDM) types.

### Table 147: Supported Data Types

<table>
<thead>
<tr>
<th>SQL Data Type</th>
<th>EDM Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>Edm.Int64</td>
</tr>
<tr>
<td>BINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>BIT</td>
<td>Edm.Boolean</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>Edm.Boolean</td>
</tr>
<tr>
<td>CHAR</td>
<td>Edm.String</td>
</tr>
<tr>
<td>DATE</td>
<td>Edm.Date</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>Edm.Decimal</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>FLOAT</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Edm.Int32</td>
</tr>
<tr>
<td>LONGVARBINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>LONGVARCHAR</td>
<td>Edm.String</td>
</tr>
<tr>
<td>REAL</td>
<td>Edm.Single</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>Edm.Int16</td>
</tr>
<tr>
<td>TIME</td>
<td>Edm.TimeOfDay</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>Edm.DateTimeOffset</td>
</tr>
<tr>
<td>TINYINT</td>
<td>Edm.Byte</td>
</tr>
<tr>
<td>VARBINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>Edm.String</td>
</tr>
</tbody>
</table>

---

For values smaller than 32 KB. Values 32 KB and larger are not supported.

Value maps to **Edm.Byte** if described as unsigned. If the value is described as signed, it maps to **Edm.SByte**.

---

**879 Progress DataDirect Hybrid Data Pipeline: User's Guide: Version 4.6**
Stored functions support

Hybrid Data Pipeline supports Oracle, DB2, and SQL Server stored functions for OData Version 4 services as described here. Stored functions can be implemented either through the Web UI or through the

• Functions that are unbound (static operations)
• Function imports
• Functions that return primitive types
• Function invocation with OData system query options $filter

Note that the following aspects of OData Version 4 functions are NOT supported.

• Functions that return complex types and entities
• Functions that are bound to entities
• Built-in functions
• Functions with OUT/INOUT parameters
• Overloaded functions
• Function invocation as part of $select
• Function invocation as part of $orderby
• Function invocation as part of parameter value
• Parameter aliases are not supported. Hence, invoking functions with function parameters as URL query parameters is not supported.

You can invoke stored functions using an OData service either independently or as part of another operation, such as a filter operation.

OData model warnings

Hybrid Data Pipeline users can get the details of the tables/columns and functions that were dropped while generating the OData Model for a given schema map of a data source. The details can be queried over the Model Status API - https://<baseUrl>/api_mgmt/datasources/<datasourceId>/model. Since the OData Model creation is asynchronous, all the warnings are stored in the ModelWarnings table and a query for the model status returns details from this table.

Possible warning messages for Tables and Columns:

• The column has an unsupported data type < >
• The column size is too long. Actual size is < > and supported size is < >
• The primary key column has an unsupported data type < >.
• No primary key has been specified for this table < >.
OData Model warnings are also generated where there is a problem mapping the user specified Functions to OData Functions. OData Model Warnings, called Operation warnings, are generated for the following scenarios:

- Model Creation with Operations for NON-ORACLE Data Sources.
- Functions that have LONG types as params or return type.
- Functions that have Non-Primitive Params or Return values.
- Functions that have OUT or INOUT Parameters.
- Functions mapped as ACTIONS in SchemaMap.
- If Stored Procedures are Mapped in SchemaMap.
- Operation does not exist.

Understanding and configuring a schema map for OData Version 4

As described in Configuring data sources for OData Version 4 connectivity on page 627, you use the Hybrid Data Pipeline dashboard’s Configure Schema editor to generate or edit a schema map. The schema map specifies the tables, or objects, and columns that will be accessible to OData clients for a particular data source definition. A schema map can only include tables from one schema. To expose tables from multiple schemas (in the same data store) or to expose multiple data stores in a single OData endpoint, you can create a data source group.

Hybrid Data Pipeline generates schema maps as a JSON string. When fetching data to satisfy requests, the Hybrid Data Pipeline OData service uses this schema to map a row in a table (or an object instance) to an entity, and to map the data in table columns (or object attributes) to entity properties. Progress recommends that you use the generated schema map. However, there are rare use cases that might require you to edit the JSON string. See JSON schema map syntax on page 882 for a description of the syntax.

Primary and foreign keys

The schema map must specify how to uniquely identify a particular record. Many data store tables already have one or more primary key columns. The Configure Schema editor checks for a primary key in the tables you select, and identifies all tables that need to have a primary key defined. If a primary key is defined on a table, the OData service uses that primary key as the unique identifier and you cannot specify another. To expose tables that do not have a primary key, you must choose one or more columns to use as a virtual primary key. Hybrid Data Pipeline automatically adds related tables for selected foreign key columns.

Note: Although the Configure Schema editor lets you specify which tables and columns to expose to OData requests, it makes no change in the underlying data source. All columns of the data source are still available to SQL queries executed from the ODBC driver, JDBC driver, or the Hybrid Data Pipeline SQL Editor regardless of whether they are exposed through OData.
Entity names
In some cases, you might want to modify the names that the Configure Schema editor assigns to an entity:

- By default, the Hybrid Data Pipeline OData service uses a plural form of the table name as the entity name. The schema generator automatically appends `es` to table names. For example, a data source table named `Customers` will become a `Customerses` entity. You might want to explicitly set the name to `Customers`.

- If you are using a data source group, table names in the member data sources can conflict. Therefore, when you create a data source group, you must assign a unique prefix to each data source definition. When this is the case, it makes sense to use the same plural name for the tables in each schema map.

Queries must have the prefix appended to the plural entity name with an underscore separator. For example, two data sources in the same group might contain a `Customer` table. In the Configure Schema editor, you could assign the plural name `Customers` to the tables in both schemas. In the data source group, you could use a prefix such as `east` for one member and `west` for the other. Query requests to the `east_Customers` entity will then go to the first data source, and requests to `west_Customers` to the second.

JSON schema map syntax
The Configure Schema editor should be used to generate the OData schema map as described in Configuring data sources for OData Version 4 connectivity on page 627. In rare cases, manual editing of the schema map might be necessary. For OData Version 4 services, an `odata_mapping_v3` format is supported.

A schema map consists of a JSON string that contains the following model elements.

```json
{
    "odata_mapping_v3": {
        "timeAsString": "boolean",
        "guidAsString": "boolean",
        "unboundNumberAsDouble": "boolean",
        "unboundNumberPrecision": "integer",
        "unboundNumberScale": "integer",
        "entityNameMode": "pluralize" or "guess" or "singularize" or "suffix",
        "singularSuffix": "suffix",
        "schemas": {
            "name": "schema_name",
            "tables": {
                "table_name": {
                    "ODataAlias": "odata_name",
                    "ODataPluralAlias": "plural_odata_name",
                    "searchMode": "none" or "begins" or "contains" or "full-text",
                    "columns": {
                        "column_name": {
                            "primaryKeyComponent": "integer",
                            "searchable": "boolean",
                            "indexType": "text_index_name",
                            "alias": "alias_name",
                            "typeInfo": {
                                "columnSize": "integer",
                                "scale": "integer",
                                "dataType": "type_name",
                                "isNullable": "boolean",
                                "isAutoIncrement": "boolean",
                                "isGenerated": "boolean"
                            }
                        },
                        "column2_name": {
                            ....
                        }
                    }
                }
            }
        }
    }
}
```
The following table lists the various elements and provides a brief description. See Schema map examples on page 886 for sample usage.

<table>
<thead>
<tr>
<th>Element name</th>
<th>Parent</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unboundNumberAsDouble</td>
<td>odata_mapping_v3</td>
<td>Indicates whether decimal columns, parameters and return values defined with no precision and scale should get automatically mapped to 'double'. This is the current default behavior for Oracle NUMBER columns declared with no precision or scale. This option allows you to map these types to OData 4 decimal type with variable scale. The default is true. When false, the OData model will describe the column or parameter as having a precision of 38 and having a scale set to &quot;variable&quot;. The defaults for precision and scale may be overridden using the unboundNumberPrecision and unboundNumberScale elements.</td>
</tr>
<tr>
<td>entityNameMode</td>
<td>odata_mapping_v3</td>
<td>Indicates the algorithm used to map the table names to the entity collection name and the entity type name. The entity collection name is normally the plural form and the entity type name is the singular form. Defaults to &quot;guess&quot;.</td>
</tr>
<tr>
<td>singularSuffix</td>
<td>odata_mapping_v3</td>
<td>The suffix to use for the singular name (entity type) that is used during the suffix naming mode. This suffix may also be used in the other naming modes in some scenarios. Default value is &quot;:_type&quot;.</td>
</tr>
<tr>
<td>unboundNumberPrecision</td>
<td>odata_mapping_v3</td>
<td>Indicates the effective precision for unbound numbers that are mapped to decimal. This option only applies when unboundNumberAsDouble is false and only applies to numbers that have been designated as being unbound. When not specified, a default of 38 is used.</td>
</tr>
<tr>
<td>unboundNumberScale</td>
<td>odata_mapping_v3</td>
<td>Indicates the effective scale for unbound numbers that are mapped to decimal. This option only applies when unboundNumberAsDouble is false and only applies to numbers that have been designated as being unbound. When not specified, a default of &quot;variable&quot; is used.</td>
</tr>
<tr>
<td>Element Name</td>
<td>Parent</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>guidAsSting</td>
<td>odata_mapping_v3</td>
<td>Indicates whether or not GUID data types are exposed as OData Edm.String. The default is false, which means that GUID data types are exposed as Edm.Guid. This option currently only applies to the SQL Server uniqueidentifier data type.</td>
</tr>
<tr>
<td>schema_name</td>
<td>None</td>
<td>The backend data source schema name. This is a required field. For data stores that do not support schemas, such as MySQL, the schemaName value should be null (&quot;schemaName&quot;: null).</td>
</tr>
<tr>
<td>excludedTables</td>
<td>schema_name</td>
<td>Comma-separated list of tables to hide from OData requests. Any tables not specified in this list, and having a primary key column will be exposed for OData requests. This optional field is used only when the tables object is missing or empty.</td>
</tr>
<tr>
<td>name</td>
<td>schema_name</td>
<td>Contains the schema_name element. This is a required property for data sources that support schemas. For data sources such as MySQL that do not support schemas, set this to &quot;null&quot; or &quot;-&quot;.</td>
</tr>
<tr>
<td>table_name</td>
<td>tables</td>
<td>The backend data source table (or object) name. This property determines the name to be used in OData requests.</td>
</tr>
<tr>
<td>tables</td>
<td>schema_name</td>
<td>Contains table_name elements describing how to expose tables through OData, and an excludedTables element listing tables that should not be exposed. If the tables object is missing or empty, all tables, except for any table in the excludedTables array, are exposed.</td>
</tr>
<tr>
<td>columns</td>
<td>table_name</td>
<td>Contains column_name elements that define the details of columns included in a table. If the columns element is missing or empty, then all columns except the ones listed in excludeColumns are exposed.</td>
</tr>
<tr>
<td>excludedColumns</td>
<td>table_name</td>
<td>Comma-separated list of columns to hide from OData requests. This optional field is used only when the columns object is missing or empty.</td>
</tr>
<tr>
<td>ODataAlias</td>
<td>table_name</td>
<td>The singular entity name to use in OData addresses for requests to this table.</td>
</tr>
<tr>
<td>ODataPluralAlias</td>
<td>table_name</td>
<td>The plural entity name to use in OData requests.</td>
</tr>
<tr>
<td>searchMode</td>
<td>table_name</td>
<td>One of: none, not searchable; begins, search for the string only at the beginning of a field; contains, search for a specific string; full-text, use the data source index. The searchMode applies to columns enabled for search.</td>
</tr>
<tr>
<td>Element name</td>
<td>Parent</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>column_name</td>
<td>columns</td>
<td>The backend data source column (or field) name. Column properties determine whether the column is part of the primary key and is searchable.</td>
</tr>
<tr>
<td>typeInfo</td>
<td>column_name</td>
<td>Advanced type information that is used to override the information that was discovered using the JDBC driver. Normally, this information should not be specified.</td>
</tr>
<tr>
<td>column_alias</td>
<td>column_name</td>
<td>The name to use as the entity property name for the column.</td>
</tr>
<tr>
<td>indexType</td>
<td>column_name</td>
<td>The model contains this element to identify the type of index when the search mode is set to Full Text. For DB2 and SQL Server, TEXT is the only valid value. For Oracle, valid values include CONTEXT and CTXCAT.</td>
</tr>
<tr>
<td>primaryKeyComponent</td>
<td>column_name</td>
<td>The data type of a column belonging to the primary key, or null. The primary key is comprised of a set of columns to be used as the primary key for a table that does not have a defined primary key. If this field is not specified or the key list is empty, the table must have a primary key defined in the database. If a primary key is defined for the table in the database and a primary key column list is also specified in the OData Schema Map parameter, the primary key defined in the database is used.</td>
</tr>
<tr>
<td>searchable</td>
<td>column_name</td>
<td>If true, the column is searchable, using the searchMode specified at the table level. If false, the column is not searchable.</td>
</tr>
<tr>
<td>isNullable</td>
<td>typeInfo</td>
<td>Indicates whether the column can have a null value. Normally drivers report this correctly. Some drivers may report a column as not nullable while null values exist in the column. In such a scenario, the isNullable could be set to true to correct this issue. Note, there could be implications on the 'create entity' behavior by changing this setting.</td>
</tr>
<tr>
<td>dataType</td>
<td>typeInfo</td>
<td>Indicates the desired data type for the column. The data type is specified as the JDBC type name.</td>
</tr>
<tr>
<td>isAutoIncrement</td>
<td>typeInfo</td>
<td>Indicates whether the column is a uniquely generated column. Setting this to true will indicate to the service that it should ignore incoming values for this column during 'create entity' and 'update/patch entity' operations.</td>
</tr>
<tr>
<td>isGenerated</td>
<td>typeInfo</td>
<td>Indicates whether the column is a generated value. If the column is generated, then the OData code will ignore incoming value for column during the 'create entity' and 'update/patch entity' requests.</td>
</tr>
<tr>
<td>Element name</td>
<td>Parent</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>dataType</td>
<td>typeInfo</td>
<td>Indicates the desired data type for the column. The data type is specified as the JDBC type name.</td>
</tr>
<tr>
<td>columnSize</td>
<td>type_info</td>
<td>Indicates the maximum precision or maximum length of the column. Some drivers may report column sizes that are not accurate or are too large.</td>
</tr>
<tr>
<td>scale</td>
<td>type_info</td>
<td>Indicates the maximum scale of the column.</td>
</tr>
</tbody>
</table>

**Schema map examples**

In the following example from an Oracle data source, both the Employees and the Departments tables are enabled for full-text search. In the Employees table, the id column is searchable. In the Departments table, the id column is searchable and the address column is not included in the model; OData requests will not return data from the address column.

```json
{
   "odata_mapping_v3": {
      "schemas": [{
         "name": "Emp",
         "tables": {
            "Employees": {
               "ODataAlias": "Employee",
               "ODataPluralAlias": "Employees",
               "searchMode": "full-text",
               "columns": {
                  "ID": {
                     "alias": "Test ID",
                     "primaryKeyComponent": 1,
                     "searchable": true,
                     "typeInfo": {
                        "dataType": "DECIMAL",
                        "columnSize": 14,
                        "isGenerated": true,
                        "isAutoIncrement": true,
                        "isNullable": false,
                        "scale": 4
                     }
                  }
               }
            }
         }
      },
      "Departments": {
         "ODataAlias": "Employee",
         "ODataPluralAlias": "Employees",
         "searchMode": "full-text",
         "columns": {
            "ID": {
               "alias": "Test Department",
               "primaryKeyComponent": 1,
               "searchable": true,
               "typeInfo": {
                  "dataType": "DECIMAL",
                  "columnSize": 24,
                  "isGenerated": true,
                  "isAutoIncrement": true,
                  "isNullable": false,
                  "scale": 4
               }
            }
         }
      }
   }
}
```
The following example uses tables in a MySQL datasource. As in the previous example, both the Employees and the Departments tables are enabled for full-text search. In the Employees table, the id column is searchable. In the Departments table, the id column is searchable and the address column is not included in the model; OData requests will not return data from the address column.

```json
{
    "odata_mapping_v3": {
        "schemas": [{
            "name": "Emp",
            "tables": {
                "Employees": {
                    "ODataAlias": "Employee",
                    "ODataPluralAlias": "Employees",
                    "searchMode": "full-text",
                    "columns": {
                        "ID": {
                            "alias": "Test ID",
                            "primaryKeyComponent": 1,
                            "searchable": true,
                            "typeInfo": {
                                "dataType": "DECIMAL",
                                "columnSize": 14,
                                "isGenerated": true,
                                "isAutoIncrement": true,
                                "isNullable": false,
                                "scale": 4
                            }
                        }
                    }
                },
                "Departments": {
                    "ODataAlias": "Employee",
                    "ODataPluralAlias": "Employees",
                    "searchMode": "full-text",
                    "columns": {
                        "ID": {
                            "alias": "Test Department",
                            "primaryKeyComponent": 1,
                            "searchable": true,
                            "typeInfo": {
                                "dataType": "DECIMAL",
                                "columnSize": 24,
                                "isGenerated": true,
                                "isAutoIncrement": true,
                                "isNullable": false,
                                "scale": 4
                            }
                        },
                        "excludedColumns": ["address"]
                    }
                }
            }
        }
    }
}
```
Structure requests for OData Version 4

OData requests to a Hybrid Data Pipeline data source must include authentication, the service root, and resource name. You can fetch single or multiple entities and related entities using entity addressing and the supported methods. While you can set some server-side behavior such as caching and paging in the data source definition, client-side options also allow you to control behaviors such as paging and response formatting.

The following are required:

• **Authentication**
  Supply credentials for Hybrid Data Pipeline and for the backend data store:
  - The Hybrid Data Pipeline user ID and password must be passed using HTTP basic authentication. The client encrypts the Hybrid Data Pipeline user ID and password in the `Authorization` header.
  - The credentials for your data store can be stored in the data source definition or passed as part of an OData request — using the `ddcloud-datasource-user` and the `ddcloud-datasource-password` headers, as described in [Data Source User Header](#) and [Data Source Password Header](#).

• **Service root and resource name**
  The location of the Hybrid Data Pipeline service and the name of the OData-enabled data source definition (case insensitive) as displayed on the OData tab of your data source definition. See [Service URI and resource path in Hybrid Data Pipeline](#) on page 891 for an example.

The following are optional:

• **Entity addressing**
  Append entity addresses to the request after the data source name. Use the plural entity name defined in the schema map. For example, the following request fetches the employee record with a primary key of 27, from the EMPLOYEES table in the `myoracletest2` data source.

  `https://<myserver>:<port>/api/odata4/myoracletest2/EMPLOYEES('27')`

  See [Service URI and resource path in Hybrid Data Pipeline](#) on page 891 and [Formulating queries with OData Version 4](#) on page 892 for details and more examples.

  where `<myserver>` is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

• **Queries and operations**
  Hybrid Data Pipeline supports OData edit, create, update and delete operations, see examples in the [Formulating queries with OData Version 4](#) on page 892 section.

**Headers**

You can use request headers to control the following service behaviors:

• Whether the response comes from cached data (if available) or from the back-end data store, as described in [Refresh Result Header](#) on page 889.

• Specify the backend data store credentials as described in [Data Source User Header](#) on page 889 and [Data Source Password Header](#) on page 889.
• The time zone to apply to DateTime values, see Timezone Header on page 889.

• Anticipate how clients will be the $top system query parameter with the Top Mode on page 890 to improve performance.

• How the service breaks up a result set into multiple responses with the OData Prefer Header - Max Page Size on page 890.

Some of these behaviors can be controlled with query parameters instead of in headers. See Custom query parameters on page 895.

Refresh Result Header

Hybrid Data Pipeline buffers the results of an OData query, allowing clients to page back and forth through the results using the $top and $skip system query parameters. The $top parameter specifies how many results to return in the first response and $skip specifies where to start in the result set to return the next set of results. When the Hybrid Data Pipeline service receives an OData query for which it has a buffered result and the $skip query parameter is either not specified or is set to zero, Hybrid Data Pipeline can page back to the beginning of the buffered result or execute a new query.

By default, Hybrid Data Pipeline treats a query where $skip is missing or set to zero as a request to re-execute the query in the backend data source. You can change default behavior in the data source definition, or in the request with the ddcloud-refresh-result header. The header value overrides the setting in the Refresh Result field of the data source definition.

<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-refresh-result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted Values</td>
<td>0, reuse cached results. 1, discard cached results and query the data store again.</td>
</tr>
<tr>
<td>Default when not specified</td>
<td>1. The service executes the query anew.</td>
</tr>
</tbody>
</table>

Data Source User Header

The credentials for the backend data source can be stored in the data source definition on the General tab. If they are not, you must supply them in requests using the ddcloud-datasource-header header.

<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-datasource-user</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default when not specified</td>
<td>The Hybrid Data Pipeline service checks the data source definition for this value.</td>
</tr>
</tbody>
</table>

Data Source Password Header

The credentials for the backend data source can be stored in the data source definition on the General tab. If they are not, you must supply them in requests using the ddcloud-datasource-password header.

<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-datasource-password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default when not specified</td>
<td>The Hybrid Data Pipeline service checks the data source definition for this value.</td>
</tr>
</tbody>
</table>

Timezone Header

To correctly process DateTime data types for clients in a different timezone than the data store, use the ddcloud-timezone header.
<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-timezone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted Values</td>
<td>A Java timezone id string.</td>
</tr>
<tr>
<td>Default when not specified</td>
<td>The timezone is taken from URL; GMT is used if timezone is not specified as a header or URL parameter</td>
</tr>
</tbody>
</table>

**Top Mode**

In some cases, the Hybrid Data Pipeline OData service can optimize requests to the backend data store when you use the `ddcloud-top-mode` to specify how a client will be using the `$top` system parameter to page through results. A value of 0 indicates that the client will use `$top` to limit the result set and will rarely request the remaining entities. A value of 1 indicates that the client will often use `$top` and `$skip` to page through results.

Hybrid Data Pipeline applies the optimization only to queries that meet the following conditions:

- Include a value for `$top`
- Do not include `$skip` or include `$skip` with a value of 0
- Do not include `$expand`
- Do not include `$count=true` with the inline count mode set to 2, which causes a fetch of all rows

When the conditions are met, Hybrid Data Pipeline will generate only a `SELECT` statement that includes the data store-specific syntax for limiting the rows returned. If the client queries the same entity collection again but specifies `$top` and `$skip` to fetch more entities, the service executes a new query. The results might contain some of the entities already received from the first request.

In the following example, the `ddcloud-top-mode` is set to 1, directing the Hybrid Data Pipeline service to fetch the complete result set and not to attempt optimization:

```
ddcloud-top-mode=1
```

<table>
<thead>
<tr>
<th>Name</th>
<th>ddcloud-top-mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted Values</td>
<td>0 indicates that the client will use <code>$top</code> to limit the result set and will rarely request the remaining entities</td>
</tr>
<tr>
<td></td>
<td>1 indicates that the client will often use <code>$top</code> and <code>$skip</code> to page through results</td>
</tr>
<tr>
<td>Default when not specified</td>
<td>0</td>
</tr>
</tbody>
</table>

**OData Prefer Header - Max Page Size**

The OData 4.0 specification defines a Prefer header, `odata.maxpagesize`, that can be used to control the page size for server-driven paging. In server-driven paging, the server returns partial results and includes a link the client can use to get the next set of results.

You can set the page size in the data source definition, on the **OData** tab, in the **Page Size** field. The request header value for `odata.maxpagesize` overrides the value specified in the data source definition. In the following example, the maximum page size is set to 4000, resulting in up to 4000 entities per page.

```
Prefer: odata.maxpagesize=4000
```
Service URI and resource path in Hybrid Data Pipeline

The service root and resource path of a request define the location of the Hybrid Data Pipeline service and the name of the OData-enabled data source definition (case insensitive). The OData tab of your data source definition provides this value in the OData Access URI field.

In the URL examples in this table,

<myserver> is the DNS name of the machine on which the Hybrid Data Pipeline server is installed.

<myds> is the name of your Hybrid Data Pipeline data source.

A request with just the Service URI and resource path returns a list of available entities in the form of the service document, the $metadata parameter returns metadata on those entities, and an address that includes the plural entity name and a primary key value returns a single entity, as shown in the following table. For additional information, see Requesting service metadata and the service document on page 875.

<table>
<thead>
<tr>
<th>Response contains:</th>
<th>Operation</th>
<th>URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>The names of all entities in the schema</td>
<td>GET</td>
<td>&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;myds&gt;</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><a href="https://mustng02:8443/api/odata4/myds/">https://mustng02:8443/api/odata4/myds/</a></td>
</tr>
<tr>
<td>The names, properties, data types, and relationships for all entities in the schema</td>
<td>GET</td>
<td>&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;myds&gt;/$metadata</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><a href="https://mustng02:8443/api/odata4/myds/$metadata">https://mustng02:8443/api/odata4/myds/$metadata</a></td>
</tr>
<tr>
<td>A single entity</td>
<td>GET</td>
<td>&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;myds&gt;/entity_plural_name('primary_key_value')</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><a href="https://mustng02:8443/api/odata4/myds/ACCOUNTS('123">https://mustng02:8443/api/odata4/myds/ACCOUNTS('123</a>')</td>
</tr>
<tr>
<td>A single entity from a particular data source in a data source group</td>
<td>GET</td>
<td>&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;myds&gt;/&lt;ds_prefix&gt;_&lt;entity_plural_name&gt;('primary_key_value')</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><a href="https://mustng02:8443/api/odata4/myds/east_ACCOUNTS('123">https://mustng02:8443/api/odata4/myds/east_ACCOUNTS('123</a>')</td>
</tr>
</tbody>
</table>

Response formatting for OData Version 4

The OData Version 4 specification supports responses only in the JSON format; therefore, Hybrid Data Pipeline supports only the JSON format when using OData Version 4.
Formulating queries with OData Version 4

Hybrid Data Pipeline supports the following:

- A set of OData system query options and custom options to control service behavior and control result pagination.

- HTTP methods for:
  - Fetching records using **GET**
  - Creating records using **POST**
  - Updating records using:
    - **PATCH**
    - **POST with the custom** `X-HTTP-Method PATCH`
  - Deleting records using:
    - **DELETE**
    - **POST with the custom** `X-HTTP-Method DELETE`

- Search for text-based columns

Query options and optimizing response times

You can refine query results using system query options, which begin with the `$` character. Add system query options to the URL to control the amount and order of data in the response. Custom query parameters on page 895 lists additional parameters specific to Hybrid Data Pipeline. In addition, topics in this section describe settings to optimize response times when paging through result sets or when using the `$count system parameter.`
The following table lists the OData query string options that Hybrid Data Pipeline supports. For detailed information about the system query options, refer to the OData specification.

**Table 148: Supported system query options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Support in Hybrid Data Pipeline</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$count</code></td>
<td>Returns the number of records in a collection, or if the collection has a filter, the number of records that match the filter.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td><strong>Note:</strong> <code>$count</code> replaces the <code>$inlinecount</code> parameter for OData v4 and higher.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>$expand</code></td>
<td>In addition to retrieving a record or collection, retrieve related records.</td>
<td>At present, supports expanding one level deep.</td>
</tr>
<tr>
<td><code>$filter</code></td>
<td>An expression or function that must evaluate to true for records that will be included in the response.</td>
<td>Supports all functionality except the following scalar functions: fractional seconds, geodistance, geointersects, geolength, isof, maxdatetime, mindatetime, totaloffsetminutes, totalseconds.</td>
</tr>
<tr>
<td><code>$orderby</code></td>
<td>Determines the values used to order a collection of records.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td><code>$top</code></td>
<td>Identifies a subset of records to return from a collection. To form this subset, select only the first N items of the set, where N is a positive integer. See Paging through results on page 895 for more information.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td><code>$skip</code></td>
<td>Identifies a subset of records to return from a collection. Define the subset by seeking N entries into the Collection and selecting only the remaining entries (starting with Entry N+1), where N is a positive integer.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td><code>$search</code></td>
<td>Searches for the specified expression in columns that are enabled for search in the schema map. Do not use <code>$search</code> and <code>$filter</code> in the same request. See Searching text-based columns on page 895 for more information.</td>
<td>Supports all standard functionality.</td>
</tr>
<tr>
<td><strong>Note:</strong> <code>$search</code> replaces the DataDirect proprietary <code>ddsearch</code> parameter for OData v4 and higher.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Improving performance when using Count

The `$count` OData system query option includes the count of the number of entities that satisfy a query in the response. The count is included in the first page in server-side paging, and in every page when the client controls paging. Possible values for the parameter include `true` and `false`:

```plaintext
inlineCountQueryOp = "$count=" "true" | "false"
```

Calculating the count for very large collections can take time. The default behavior for Hybrid Data Pipeline differs for relational and cloud data sources:

- For relational data stores, by default, Hybrid Data Pipeline sends a separate query to get the count before requesting the records. This behavior tends to result in a quicker response for the first page of results. However, it requires two queries to be executed rather than one. And, in some data sources, the count(*) aggregate is not efficiently implemented.

- For cloud-based data stores, by default, Hybrid Data Pipeline fetches the entire result before returning the first page. For small results, this approach will always be faster. However, this approach may have longer initial response time for the first page if the result is large.

This behavior can be changed in the data source definition, as described in Configuring data sources for OData Version 4 connectivity on page 627 or by using the `$count` parameter. With a value of `true`, Hybrid Data Pipeline will include the count in the response. For example:

```plaintext
```

With a value of `false`, Hybrid Data Pipeline avoids obtaining a count and avoiding the associated overhead. For example:

```plaintext
```
Paging through results

Hybrid Data Pipeline divides results that exceed a threshold into multiple pages. For OData queries, you can use server-side or client-side pagination:

- By default, Hybrid Data Pipeline divides OData responses with a maximum of 2000 top-level entities per response. If the response is larger than 2000 entities, the first page contains the first 2000 entities and contains a next link at the end of the response. The next link contains the URL to fetch the next page of results. Next link URLs should be passed back without modification. You can modify the maximum number of entities returned in a page by setting the OData **PageSize** data source parameter as described in Configuring data sources for OData Version 4 connectivity on page 627.

- Client-side pagination is controlled by both the client and the Hybrid Data Pipeline OData service. Requests can specify a particular page size with the $top query parameter and can navigate through the pages by specifying different values for the $skip query parameter. The **Top Mode** setting allows the Hybrid Data Pipeline service to optimize queries in certain situations. You can set the **Top Mode** in the data source definition or use the *ddcloud-top-mode* header in requests to inform the service of how the client uses $top. See Configuring data sources for OData Version 4 connectivity on page 627 and Top Mode on page 890 for more information.

For example, the following URL requests Employees entities in pages of 100.

https://<myserver>:<port>/api/odata4/OracleOPTest/EMPLOYEES?$top=100&$skip=0

To fetch the next page, increment the $skip parameter by the page size.

https://<myserver>:<port>/api/odata4/OracleOPTest/EMPLOYEES?$top=100&$skip=100

The client can request any page size it needs. However, the Hybrid Data Pipeline connectivity service might return fewer entities than were requested. In this case, the response will contain a next link, as with server-side paging. The client should use the next link(s) to get all of the results before requesting the next page.

Custom query parameters

For OData Version 4, Hybrid Data Pipeline OData service provides the following custom query parameter.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>timezone</td>
<td>A Java timezone id string. If the client timezone differs from that of the Hybrid Data Pipeline service, specifying the timezone might be necessary to correctly process DateTime values. The timezone can also be specified as header. See OData Headers for more information.</td>
<td>When not specified in the URL or as a header, defaults to GMT.</td>
</tr>
</tbody>
</table>

Searching text-based columns

Different data store types support different levels of indexing and searching. Indexing increases the efficiency of searches in tables with many records. Querying to find particular values can be expensive when the search must span many columns and many records. To improve performance, you can restrict searches to particular text-based columns by using the search functionality in your queries. For OData version 4, searches are executed using the $search system query option. To search across all columns in the schema, even those not enabled in the schema map for searching, you can use OData $filter.
This release supports use of $search for all data store types, and full-text search taking advantages of indexes in the following data source types:

- **DB2 on Linux, UNIX, and Windows** — Each column to be searched must have a separate full text index, the full text services must be running, and the database must be enabled for full text. See the DB2 documentation for more information.

- **Oracle** — Each column to be searched must have a separate full text index, the full text services must be running, and the database must be enabled for full text. See the Oracle documentation for more information.

- **Microsoft SQL Server** — Each column to be searched must have a separate full text index and the full text index engine must be running. See the Microsoft documentation for more information.

To use text search with OData version 4:

1. For data stores that support full-text search, make sure that the underlying data store is indexed and is up to date with the current schema.

2. Enable search for the indexed columns in the Hybrid Data Pipeline data source schema map, as described in Configuring data sources for OData Version 4 connectivity on page 627 and selecting Full Text as the search type.

3. Use the $search query option with a search string. For details, refer to the OData Version 4.0 Specification.

Hybrid Data Pipeline treats multiple terms by using a logical and. For example, a search for Sales & Marketing returns records that contain both the word Sales and the word Marketing, the ampersand is ignored. The case-sensitivity of the search string depends on the underlying data source.

**Note:** The hash (#) character is not allowed in a search expression. To use the hash character in a search expression, it will need to be percent encoded.

### Fetching records and collections

As shown in the following table, use the plural entity name with the GET method to fetch metadata, a single entity, an entity’s property, or a collection of entities. When using a data source group, prepend the entity name with the appropriate data source prefix. See URI conventions for addressing resources, entities, and related entities in Section 2 of the OData version 4 specification.

<table>
<thead>
<tr>
<th>To fetch:</th>
<th>Method:</th>
<th>URI:</th>
</tr>
</thead>
<tbody>
<tr>
<td>A single record</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_singular_name&gt;('&lt;primary_key_value&gt;')</code></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><code>https://myserver:8080/api/odata4/MySFDataSource/ACCOUNTS('1')</code></td>
</tr>
<tr>
<td>To fetch:</td>
<td>Method:</td>
<td>URI:</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>----------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>The value of a single field from a single record</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_singlar_name&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(<code>&lt;primary_key_value&gt;</code>)/&lt;column_name&gt;/$value</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><a href="https://myserver:8080/api/odata4/MySFDataSource/ACCOUNTS('1')/NAME/$value">https://myserver:8080/api/odata4/MySFDataSource/ACCOUNTS('1')/NAME/$value</a></td>
</tr>
<tr>
<td>A collection of records*</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_plural_name&gt;</code></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><a href="https://myserver:8080/api/odata4/MySFDataSource/ACCOUNTS">https://myserver:8080/api/odata4/MySFDataSource/ACCOUNTS</a></td>
</tr>
<tr>
<td>A count of the records in a collection</td>
<td>GET</td>
<td><code>&lt;service_root&gt;/&lt;data_source_name&gt;/&lt;entity_plural_name&gt;/$count</code></td>
</tr>
<tr>
<td>Example:</td>
<td></td>
<td><a href="https://myserver:8080/api/odata4/MySFDataSource/ACCOUNTS/$count">https://myserver:8080/api/odata4/MySFDataSource/ACCOUNTS/$count</a></td>
</tr>
</tbody>
</table>

*A single request can only fetch one collection.

**Creating, editing, and deleting records**

Create records using the POST method. Update records using the PATCH method or the POST method with the custom header, X-HTTP-Method, with a value of patch. Delete records using the DELETE method or the POST method with X-HTTP-Method with a value of DELETE. A request should include:

- Your Hybrid Data Pipeline account credentials.
- If the backend data source credentials are not stored in the Data Source definition, the ddcloud-datasource-user and the ddcloud-datasource-password headers.
- The resource URL appropriate for the operation:
  - To create a record, include the plural entity name and supply property values in the body.
  - To update a record, include the plural entity name and the primary key value.
  - To delete a record, include the plural entity name and the primary key value.

To create or update, supply property values, the Content-Type header must specify one of the following supported content types:

application/json
application/json;charset=UTF-8
If the Content-Type header is not supplied, Hybrid Data Pipeline interprets the body as the JSON format encoded using the UTF-8 character set.

**Create example**

When supplying property values, include required columns (except for those with default values or set automatically by the data store). The following screen shows a POST request in Postman to create an ACCOUNT entity in a Salesforce data store. To formulate the request:

- **The header** `Content-Type` **has the value** `application/json`.
- **The URL includes:**
  - The service root, `<myserver>:<port>/api/odata4`.
  - The `DataSource` definition name, `sfds`.
  - The plural entity name, `ACCOUNTS`.
- **The body includes:**
  - Fields that were copied from the response of a GET request that fetched a single account record.
  - No value was supplied for `ROWID`, the primary key, because Salesforce generates the value automatically.

The following lines in the response show that the new record was successfully created:

For more details on creating records and an example in JSON format, see [HTTP POST (create)](page906) on page 906

**Delete example**

To delete a record, use HTTP DELETE or the POST request with the custom `X-HTTP-Method` header value of DELETE. Supply the primary key of the record to delete. The following screen shows using an HTTP DELETE request in Postman to delete a record from a Salesforce data store.
To formulate the request:

- **The Content-Type header value is** `application/json`.
- **The resource URL includes:**
  - **The service root,** `<myserver>:<port>/api/odata4`.
  - **The Data Source definition name,** `sfds`.
  - **The plural entity name,** `ACCOUNTS` followed by the primary key.
- **The body of the request is empty.**

The following screen shows the result of executing the request. **The Status of 204 No Content indicates** that the record was successfully deleted.

---

**Update example**

To update a record, use a PATCH request or a POST request with the custom `X-HTTP-Method` header. Supply the primary key in the resource URL and the property value(s) for the column(s) to update in the body. The following screen shows a PATCH request in Postman to update an account name from **Hot Diggity Dog** to **Hot Diggity Dogs** in a Salesforce data store. To formulate the request:

- **The Content-Type header value is** `application/json`.
- **The URL includes:**
  - **The service root,** `<myserver>:<port>/api/odata4`.
  - **The Data Source definition name,** `sfds`.
  - **The plural entity name,** `ACCOUNTS` followed by the primary key, `0011I000002ifiUQAQ` (which is cut off in the screen shot).
- **The body includes:**
  - A **value of Hot Diggity Dogs for the SYS_NAME field**.
The **Status** value **204 No Content** shown in the screen above indicates that the name was successfully updated. A fetch of the record confirms the update to Hot Diggity Dogs as shown below:

```
{  
  "@odata.context": "https://server.myservice/api/odata2/efds/8metdata8ACC0085",  
  "value": [  
    {  
        "RNFD": "88811000000000000000",  
        "SYS_NAME": "Hot Diggity Dogs",  
        "BILLING ADDRESS": "38888 Brody In",  
        "BILLING CITY": "Austin",  
        "BILLING STATE": "TX",  
        "ACCOUNT_NUMBER": "123456"  
    }  
  ]
}
```

For more information on updating, see [HTTP PATCH or POST and PATCH (update)](page) on page 905

**Batch Requests**

Hybrid Data Pipeline supports batch functionality for data sources using OData Version 4. Batch requests allow you to submit multiple operations in the form of a single endpoint request. Operations are submitted in the HTTP request payload and can include individual requests and change sets. Refer to the [OData Version 4.0 Specification](page) for details on formatting a batch request.

The OData 4 specification requires that all operations for a change set should fail if a single operation fails. However, for data stores that do not support transactions, Hybrid Data Pipeline permits some of the operations to successfully complete when an error occurs. This behavior allows for batch requests to be supported when transactions are not, but may negatively impact data integrity should an error occur. If you are connecting using one of the following data stores, do not use batch operations if a high-level of data integrity is required.

- Apache Hadoop Hive
- FinancialForce
- Google Analytics
- Oracle Analytics
- Oracle Sales Cloud
- Oracle Service Cloud
Navigating relationships

Most data source types supported by Hybrid Data Pipeline use relationships to define associations between tables or objects. In a relational data source, foreign key columns reference the primary key column of the related table. When you configure a schema map for a data source that contains relationships, Hybrid Data Pipeline maps them as OData relationships. The OData model (returned via $metadata) identifies these as Navigation Properties. OData provides the following ways to access related entities:

- Resource Path navigation — fetch all related records or a specific record or property of that record.
- $expand — return links to the related records for a specific entity.
- $ref (not currently supported)\(^\text{21}\) — fetch all records for an entity and embed all related records in the response.

Hybrid Data Pipeline currently supports navigating relationships with Resource Path navigation and the $expand property. The topics in this section use an example of customers and orders with the following model:

```
Customer ---> Order ---> OrderItem
|    ---> Contact
```

**Resource Path Navigation**

Resource path navigation allows a query to reference a related entity from a parent or child entity. For example, with the following table structure, a customer’s orders can be referenced from a Customer record, as shown below.

```
Customer ---> Order ---> OrderItem
|    ---> Contact
```

**List the orders for a particular customer**

```
https://<myserver>:<port>/api/odata4/OracleDS/Customer('3')/Orders
```

**List the order items for a particular order for customer 3**

```
https://<myserver>:<port>/api/odata4/OracleDS/Customer('3')/Order('5')/OrderItems
```

\(^{21}\) The $links construct was replaced by the $ref in OData version 4. However, $ref is not currently supported by Hybrid Data Pipeline.
Access a particular order item

https://<myserver>:<port>/api/odata4/OracleDS/Customers('3')/Orders('5')/OrderItems('6')

Access a particular property

https://<myserver>:<port>/api/odata4/OracleDS/Customers('3')/Name

https://<myserver>:<port>/api/odata4/OracleDS/Customers('3')/Orders('5')/OrderItems('6')/ItemName

$expand query parameter

The examples in this topic use the following table structure:

<p>| Customer ---&gt; Order ---&gt; OrderItem |
| ---|---|---|---|</p>
<table>
<thead>
<tr>
<th>---</th>
<th>---</th>
<th>---</th>
</tr>
</thead>
</table>

The $expand system query parameter allows the related information to be embedded in the response of the parent or child entity. For example, you can obtain a list of customers with a list of all of their orders by issuing the query:

https://<myserver>:<port>/api/odata4/OracleDS/Customers?$expand=Orders

Each customer entity in the response contains the list of order entities belonging to that customer embedded in the customer entity. Multiple tables can be expanded. The following query returns the list of customer entities; embedded in each customer entity is the list of their orders and the list of contacts for that customer.

https://<myserver>:<port>/api/odata4/OracleDS/Customers?$expand=Orders, Contacts

Hybrid Data Pipeline currently only allows expanding to one level deep. For example, the following multi-level query, which attempts to expand orders and order items for a customer, is not currently supported:


For OData 4 users, the results of the $expand query parameter can be refined by using $select, *, $filter, and $top system query options. For example, the following query returns the entity Price in addition to the entities related to Orders.


Method Reference for OData Version 4

The Hybrid Data Pipeline OData service interface supports GET, PATCH, POST, POST/PATCH and POST/DELETE HTTP methods. Each operation acts on the resource specified in the URL.

The POST request to create or update an entity should include a Content-Type header specifying the format of the request payload. The Hybrid Data Pipeline OData API recognizes the following content types:

- application/json
- application/json;charset=UTF-8

If the Content-Type header is not supplied, Hybrid Data Pipeline interprets the body as the JSON format encoded using the UTF-8 character set.
**Supported OData API Operations**

The following table shows the operations that can be performed and their associated URLs. Refer to the specified section for detailed descriptions for these operations. Query the data source name to get a list of the valid entities.

In this table, `<myserver>` is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

**Note:** Unless the ports 80 and 443 are redirected to 8080 and 8443 respectively, you must specify `<myserver>:<port>`.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fetch Data from an OData Service</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;data-source-name&gt;&lt;entity-plural-name&gt;</td>
</tr>
<tr>
<td>Create an Entity</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;data-source-name&gt;&lt;entity-plural-name&gt;</td>
</tr>
<tr>
<td>Update an Entity</td>
<td>PATCH</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;data-source-name&gt;&lt;entity-plural-name&gt;('primary-key')</td>
</tr>
<tr>
<td>OR</td>
<td>POST</td>
<td>X-HTTP-Method:PATCH</td>
</tr>
<tr>
<td>Delete an Entity</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/odata4/&lt;data-source-name&gt;&lt;entity-plural-name&gt;('primary-key')</td>
</tr>
<tr>
<td>OR</td>
<td>POST</td>
<td>X-HTTP-Method:DELETE</td>
</tr>
</tbody>
</table>

**HTTP GET**

**Purpose**

Fetch an entity, collection of entities, or a property of an entity. The authenticated user must be the owner of the data source requested. If the authenticated user is not the owner of the data source, a "data source not found" error is returned.

**URL**

https://<myserver>:<port>/api/odata4/<resource path>

where

where `<myserver>` is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.
<resource path> is the address of an entity, entity collection, or a property of an entity. See Service URI and resource path in Hybrid Data Pipeline on page 891 for more information on addressing entities.

Method
GET

Response
A JSON representation of the entity, entity collection, or entity property specified in the URL.

Authentication
Basic Authentication using the Hybrid Data Pipeline account user ID and password.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must use same credentials used to create the data source definition.

See also
Creating an Entity on page 906

HTTP DELETE or POST and DELETE

Purpose
HTTP DELETE deletes a specified entity. Alternatively, you can use HTTP POST and specify DELETE as the value of the X-HTTP-Method header. The body of the request must be empty and the URL should not contain parameters.

URL
https://<myserver>:<port>/api/odata4/<entity collection>/<entity instance>
where <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

Method
DELETE | POST with a X-HTTP-Method header value of DELETE.

Response Status
If the entity is successfully deleted, the OData service returns a status of 204 No Content.

Authentication
Basic Authentication using Login ID and Password.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must use same credentials used to create the data source definition.
Sample Requests

DELETE https://service.myserver.com:8080/api/odata4/Customers(123)

POST https://service.myserver.com:8080/api/odata4/Customers(123)
X-HTTP-Method: DELETE

HTTP PATCH or POST and PATCH (update)

Purpose
HTTP PATCH updates an entity. You can also use HTTP POST and specify PATCH as the value of the custom X-HTTP-Method header. The body of the request should contain an entity description of the properties of the entity to be changed.

Note: Hybrid Data Pipeline supports neither HTTP UPDATE nor OData PUT semantics.

URL
https://<myserver>:<port>/api/odata4/<entity collection>/<entity instance>
where <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

Method
PATCH | POST with a X-HTTP-Method header value of PATCH.

Syntax
The request uses the following formats:

PATCH https://myserver:8080/api/odata4/Customers(123)
accept: application/<content-type>[,<content-type>]

POST https://myserver:8080/api/odata4/Customers(123)
accept: application/<content-type>[,<content-type>]
X-HTTP-Method: PATCH

Response
None.

Response Status
If the entity is successfully updated, the OData service returns a 204 No Content status.

Restrictions
You cannot update a property that is part of the primary key; if you supply a value, Hybrid Data Pipeline will ignore it. If a property in the entity description does not correspond to a property in the entity, then an error with a 400 Bad Request status is returned.

An HTTP request with the method set to MERGE is not supported and will return a 405 Method Not Supported response status.
Authentication
Basic Authentication using Login ID and Password. The authenticated user must use same credentials used
to create the data source definition.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the data source.

HTTP POST (create)

Purpose
Create an entity in an existing entity collection — a table or object in the underlying data store. The body of
the POST request describes the entity to be created and can be specified in the JSON OData format. Use the
Content-Type header to specify the format.

Entity descriptions include the following:
• Values for all required properties, which include those that map to an updateable column in the data store
  that is defined as NOT NULL, that does not have a default value, and is not automatically generated by the
data source.
• Optionally, include values for property values that cannot be updated. However, in this release, Hybrid Data
  Pipeline ignores these values.
• Optionally, specify values for navigation properties to create a relationship with other records.

URL
https://<myserver>:<port>/api/odata4/<data source name>/<entity collection path>
where <myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

Method
POST

Response
The body of the response contains the value of the new entity in the same format in which the entity definition
was provided in the request. The entity value returned includes the correct values for any computed or
auto-generated properties, and the Location header. The value of the Location header is the URL of the entity
inserted. For example, the location header for the entity created in the preceding example may have the value.

https://myserver:8080/api/odata4/myoracle/Products(10)

Response Status
If the entity is created successfully, the OData service returns a 201 Created status. The body of the response
contains the value of the new entity in the same format as the entity definition provided in the request. The
entity value returned includes the correct values for any computed or auto-generated properties, as well as the
Location header, which contains the URL of the entity created.

If the value for a required property is omitted from the entity description, the OData service returns a 400 Bad
Request response. The message provides an indication of which required property was not specified.
Authentication
Basic Authentication using the Hybrid Data Pipeline user ID and password. The credentials used for the request must be the same credentials used to create the Data Source definition.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must use same credentials used to create the Data Source definition.

Sample Request Payload
The following example creates a new Product entity in an Oracle data source.

```json
POST https://myserver:8080/api/odata4/myoracle/Products
{
  "ID" : 10, "Name" : "Hosta",
  "Description" : "With new features",
  "ReleaseDate" : "/Date(1436342315266)\",
  "Rating" : 1,
  "Price" : "1.23"
}
```
Querying data stores with SQL

For details, see the following topics:

• Querying data stores with SQL
• Supported data types
• Supported scalar functions
• Using Salesforce reports
• Supported SQL and Extensions
• Catalog tables
• Error messages
• Performance tuning

Querying data stores with SQL

The Hybrid Data Pipeline connectivity service supports a variety of data types, SQL commands and extensions. In addition, the service creates catalog tables to store meta-data and the results of certain functions. Applications can access this information through APIs and you can use supported functions in the SQL Editor when logged into your Hybrid Data Pipeline account.

See the following topics for details:

• Supported data types on page 910
Chapter 8: Querying data stores with SQL

- Supported scalar functions on page 940
- Using Salesforce reports on page 966
- Supported SQL and Extensions on page 967
- Catalog tables on page 1002
- Error messages on page 1006
- Performance tuning on page 1025

**Supported data types**

Data types differ depending on the data store you are accessing and whether your application connects using ODBC, JDBC, or OData.

**Note:** Salesforce data stores include Salesforce, Veeva CRM, FinancialForce, and ServiceMax.
Entity Data Model (EDM) types for OData Version 4

To support communication between an OData client and a backend data store, Hybrid Data Pipeline uses a schema map to convert data to the appropriate type for the receiver. You configure the schema map in Hybrid Data Pipeline where it is generated as a JSON string with the following OData Entity Data Model (EDM) types.

Table 149: Supported Data Types

<table>
<thead>
<tr>
<th>SQL Data Type</th>
<th>EDM Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>Edm.Int64</td>
</tr>
<tr>
<td>BINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>BIT</td>
<td>Edm.Boolean</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>Edm.Boolean</td>
</tr>
<tr>
<td>CHAR</td>
<td>Edm.String</td>
</tr>
<tr>
<td>DATE</td>
<td>Edm.Date</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>Edm.Decimal</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>FLOAT</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Edm.Int32</td>
</tr>
<tr>
<td>LONGVARBINARY(^{22})</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>LONGVARCHAR(^{22})</td>
<td>Edm.String</td>
</tr>
<tr>
<td>REAL</td>
<td>Edm.Single</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>Edm.Int16</td>
</tr>
<tr>
<td>TIME</td>
<td>Edm.TimeOfDay</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>Edm.DateTimeOffset</td>
</tr>
<tr>
<td>TINYINT</td>
<td>Edm.Byte</td>
</tr>
<tr>
<td>VARBINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>Edm.String</td>
</tr>
</tbody>
</table>

\(^{22}\) For values smaller than 32 KB. Values 32 KB and larger are not supported.

\(^{23}\) Value maps to EDM.Byte if described as unsigned. If the value is described as signed, it maps to EDM.SByte.
Entity Data Model (EDM) types for OData Version 2

To support communication between an OData client and a backend data store, Hybrid Data Pipeline uses a schema map to convert data to the appropriate type for the receiver. You configure the schema map in Hybrid Data Pipeline where it is generated as a JSON string with the following OData Entity Data Model (EDM) types.

### Table 150: Supported Data Types for OData version 2

<table>
<thead>
<tr>
<th>SQL Data Type</th>
<th>EDM Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>Edm.Int64</td>
</tr>
<tr>
<td>BINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>BIT</td>
<td>Edm.Boolean</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>Edm[Boolean]</td>
</tr>
<tr>
<td>CHAR</td>
<td>Edm.String</td>
</tr>
<tr>
<td>DATE</td>
<td>Edm.DateTime</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>Edm.Decimal</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>FLOAT</td>
<td>Edm.Double</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Edm.Int32</td>
</tr>
<tr>
<td>LONGVARBINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>LONGVARCHAR</td>
<td>Edm.String</td>
</tr>
<tr>
<td>REAL</td>
<td>Edm.Single</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>Edm.Int16</td>
</tr>
<tr>
<td>TIME</td>
<td>Edm.DateTime</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>Edm.DateTime (no timezone)</td>
</tr>
<tr>
<td></td>
<td>Edm.DateTimeOffset (with timezone)</td>
</tr>
<tr>
<td>TINYINT</td>
<td>Edm.SByte</td>
</tr>
<tr>
<td>VARBINARY</td>
<td>Edm.Binary</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>Edm.String</td>
</tr>
</tbody>
</table>

1 For values smaller than 32 KB. Values 32 KB and larger are not supported.
Amazon Redshift data types

The following table shows how the Amazon Redshift data types are mapped to the standard JDBC and ODBC data types.

<table>
<thead>
<tr>
<th>Amazon Redshift data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>CHARACTER</td>
<td>CHAR</td>
<td>SQL_CHAR</td>
</tr>
<tr>
<td>CHARACTER VARYING</td>
<td>VARCHAR or LONGVARCHAR</td>
<td>SQL_VARCHAR or SQL_LONGVARCHAR</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
</tr>
<tr>
<td>DOUBLE PRECISION</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>NUMERIC</td>
<td>NUMERIC</td>
<td>SQL_NUMERIC</td>
</tr>
<tr>
<td>REAL</td>
<td>REAL</td>
<td>SQL_REAL</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
</tbody>
</table>

Apache Hive data types

The following table shows how the Apache Hive data types are mapped to the standard data types for ODBC and JDBC.

**Note:** When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, the varchar(max) type is mapped to the Unicode type SQL_WVARCHAR.

<table>
<thead>
<tr>
<th>Apache Hive type</th>
<th>JDBC type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARRAY</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_WVARCHAR(12)</td>
</tr>
</tbody>
</table>

24 Numeric maps to SQL_NUMERIC if the precision of the NUMERIC is less than or equal to 38. If the precision is greater than 38, the driver maps the column to SQL_VARCHAR.
### Autonomous REST Connector data types

The following table shows supported REST API data types and how they are mapped to the standard data types for ODBC and JDBC.

<table>
<thead>
<tr>
<th>Apache Hive type</th>
<th>JDBC type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>BIGINT</td>
<td>SQL_BIGINT(type-5)</td>
</tr>
<tr>
<td>BINARY</td>
<td>VARBINARY</td>
<td>SQL_VARBINARY(-3)</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>BOOLEAN</td>
<td>SQL_BIT(type-7)</td>
</tr>
<tr>
<td>CHAR</td>
<td>CHAR</td>
<td>SQL_WCHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>SQL_TYPE_DATE(91) or SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>FLOAT</td>
<td>REAL</td>
<td>SQL_REAL(7)</td>
</tr>
<tr>
<td>INT</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>MAP</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_WVARCHAR(12)</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT(5)</td>
</tr>
<tr>
<td>STRING</td>
<td>VARCHAR or LONGVARCHAR$^{25}$</td>
<td>SQL_WVARCHAR(-9) or SQL_WVARCHAR(12) or SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>STRUCT</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_WVARCHAR(12)</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>TINYINT</td>
<td>TINYINT</td>
<td>SQL_TINYINT(-6)</td>
</tr>
<tr>
<td>UNION</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_WVARCHAR(12)</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_WVARCHAR(12)</td>
</tr>
</tbody>
</table>

---

$^{25}$ If the StringDescribeType parameter is set to `varchar` (the default), this data type maps to VARCHAR. If set to `longvarchar`, this data type maps to LONGVARCHAR.
### Table 153: REST API Data Types

<table>
<thead>
<tr>
<th>REST API Data Type</th>
<th>JDBC Data Type</th>
<th>ODBC Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BigInt</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>Binary</td>
<td>BINARY</td>
<td>SQL_BINARY</td>
</tr>
<tr>
<td>Bit</td>
<td>BIT</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>Boolean</td>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>Char</td>
<td>CHAR</td>
<td>SQL_CHAR</td>
</tr>
<tr>
<td>Date</td>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
</tr>
<tr>
<td>Decimal</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>Double</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>Float</td>
<td>DATETIME</td>
<td>SQL_FLOAT</td>
</tr>
<tr>
<td>GUID</td>
<td>GUID</td>
<td>SQL_GUID</td>
</tr>
<tr>
<td>Integer</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>JSON</td>
<td>JSON</td>
<td>SQL_VARCHAR</td>
</tr>
<tr>
<td>LongVarBinary</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>LongVarChar</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR</td>
</tr>
<tr>
<td>NVarchar</td>
<td>NVARCHAR</td>
<td>SQL_UNICODE_VARCHAR</td>
</tr>
<tr>
<td>SmallInt</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT</td>
</tr>
<tr>
<td>Time</td>
<td>TIME</td>
<td>SQL_TYPE_TIME</td>
</tr>
<tr>
<td>TimeWithTimeZone</td>
<td>TIMEWITHTIMEZONE</td>
<td>SQL_TYPE_TIME</td>
</tr>
<tr>
<td>Timestamp</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>TimestampWithTimeZone</td>
<td>TIMESTAMPWITHTIMEZONE</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>TinyInt</td>
<td>TINYINT</td>
<td>SQL_TINYINT</td>
</tr>
<tr>
<td>VarBinary</td>
<td>VARBINARY</td>
<td>SQL_VARBINARY</td>
</tr>
<tr>
<td>VarChar</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR</td>
</tr>
<tr>
<td>VarCharIgnoreCase</td>
<td>VARCHARIGNORECASE</td>
<td>SQL_VARCHAR</td>
</tr>
</tbody>
</table>
DB2 data types

The following table shows how the DB2 data types are mapped to the standard data types for ODBC and JDBC.

Note: When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, the varchar(max) type is mapped to the Unicode type SQL_WLONGVARCHAR.

Table 154: DB2 data types

<table>
<thead>
<tr>
<th>DB2 data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT 26</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>BINARY 26</td>
<td>BINARY</td>
<td>SQL_BINARY(-2)</td>
</tr>
<tr>
<td>BLOB 27</td>
<td>BLOB</td>
<td>SQL_LONGVARCHAR(-4)</td>
</tr>
<tr>
<td>CHAR</td>
<td>CHAR</td>
<td>SQL_WCHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>CHAR() FOR BIT DATA</td>
<td>BINARY</td>
<td>SQL_BINARY(-2)</td>
</tr>
<tr>
<td>CLOB</td>
<td>CLOB</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE or TIMESTAMP 28</td>
<td>SQL_TYPE_DATE(91) or SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>DBCLOB</td>
<td>CLOB or NCLOB 29</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>LONG VARCHAR</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
</tbody>
</table>

26 Supported only for DB2 V9.1 for z/OS.
27 Supported only for DB2 V8.1 and higher for Linux/UNIX/Windows, DB2 for z/OS, and DB2 for i V5R2.
28 For DB2 V9.7 for Linux/UNIX/Windows with the Oracle compatibility feature enabled, the Date type maps to the JDBC TIMESTAMP type.
29 When JDBCBehavior=0, the data type depends on the JVM used by the application. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and higher, the second value applies.
Greenplum data types

The following table shows how the Greenplum data types are mapped to the standard data types for ODBC and JDBC.

Note: When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, the varchar(max) type is mapped to the Unicode type SQL_WLONGVARCHAR.

Table 155: Greenplum data types

<table>
<thead>
<tr>
<th>Greenplum data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>Greenplum data type</td>
<td>JDBC data type</td>
<td>ODBC data type</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>BIGSERIAL</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>BIT</td>
<td>BIT or BINARY</td>
<td>SQL_BIT(-7) or SQL_BINARY(-2)</td>
</tr>
<tr>
<td>BIT VARYING</td>
<td>BINARY</td>
<td>SQL_BINARY(-2)</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>BOOLEAN</td>
<td>SQL_BIT(-7)</td>
</tr>
<tr>
<td>BYTEA</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(4)</td>
</tr>
<tr>
<td>CHARACTER</td>
<td>CHAR</td>
<td>SQL_WCHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>CHARACTER VARYING</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>SQL_TYPE_DATE(91)</td>
</tr>
<tr>
<td>DOUBLE PRECISION</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>NUMERIC</td>
<td>NUMERIC</td>
<td>SQL_NUMERIC(2)</td>
</tr>
<tr>
<td>REAL</td>
<td>REAL</td>
<td>SQL_REAL(7)</td>
</tr>
<tr>
<td>SERIAL</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT(5)</td>
</tr>
<tr>
<td>TEXT</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>TIME</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIME(93)</td>
</tr>
<tr>
<td>TIME WITH TIMEZONE</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>TIMESTAMP WITH TIMEZONE</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
</tbody>
</table>

**Informix data types**

The following table shows how the Informix data types are mapped to the standard data types for ODBC and JDBC.
## Supported data types

<table>
<thead>
<tr>
<th>Informix</th>
<th>JDBC</th>
<th>ODBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOB</td>
<td>BLOB</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>BIT</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>BYTE</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>CHAR</td>
<td>CHAR</td>
<td>SQL_CHAR</td>
</tr>
<tr>
<td>CLOB</td>
<td>CLOB</td>
<td>SQL_LONGVARCHAR or SQL_WLONGVARCHAR</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
</tr>
<tr>
<td>DATETIME YEAR TO SECOND</td>
<td>TIME</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>DATETIME YEAR TO DAY</td>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
</tr>
<tr>
<td>DATETIME HOUR TO SECOND</td>
<td>TIME</td>
<td>SQL_TYPE_TIME</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>FLOAT</td>
<td>FLOAT</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>INT8</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>MONEY</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>NCHAR</td>
<td>CHAR or NCHAR</td>
<td>SQL_CHAR or SQL_WCHAR</td>
</tr>
<tr>
<td>NVARCHAR</td>
<td>VARCHAR or NVARCHAR</td>
<td>SQL_VARCHAR</td>
</tr>
<tr>
<td>SERIAL</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>SERIAL8</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>SMALLFLOAT</td>
<td>REAL</td>
<td>SQL_REAL</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT</td>
</tr>
<tr>
<td>TEXT</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR or SQL_WLONGVARCHAR</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR or SQL_WVARCHAR</td>
</tr>
</tbody>
</table>
Microsoft Dynamics CRM Online data types

In communication between your application and the data store, data is mapped several times to the type appropriate for sending and receiving components. The following Microsoft Dynamics CRM Online attribute types are supported.
### Table 157: Supported data types

<table>
<thead>
<tr>
<th>Data Store data type</th>
<th>Intermediary data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>boolean</td>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>CUSTOMER</td>
<td>string</td>
<td>CHAR</td>
<td>SQL_CHAR or SQL_WCHAR&lt;sup&gt;30&lt;/sup&gt;</td>
</tr>
<tr>
<td>DATETIME</td>
<td>datetime</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>decimal</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>double</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>INTEGER</td>
<td>int</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>LOOKUP</td>
<td>string</td>
<td>CHAR</td>
<td>SQL_CHAR or SQL_WCHAR&lt;sup&gt;30&lt;/sup&gt;</td>
</tr>
<tr>
<td>MANAGEDPROPERTY</td>
<td>boolean</td>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>MEMO</td>
<td>string</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR or SQL_WLONGVARCHAR&lt;sup&gt;31&lt;/sup&gt;</td>
</tr>
<tr>
<td>MONEY</td>
<td>decimal</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>OWNER</td>
<td>string</td>
<td>CHAR</td>
<td>SQL_CHAR or SQL_WCHAR&lt;sup&gt;30&lt;/sup&gt;</td>
</tr>
<tr>
<td>PICKLIST</td>
<td>int</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>STATE</td>
<td>int</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>STATUS</td>
<td>int</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>STRING</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR or SQL_WVARCHAR&lt;sup&gt;31&lt;/sup&gt;</td>
</tr>
<tr>
<td>UNIQUEIDENTIFIER</td>
<td>string</td>
<td>CHAR</td>
<td>SQL_CHAR or SQL_WCHAR&lt;sup&gt;30&lt;/sup&gt;</td>
</tr>
<tr>
<td>VIRTUAL</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR or SQL_WVARCHAR&lt;sup&gt;31&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

1. The driver returns the WCHAR types when the Hybrid Data Pipeline ODBC driver's connection option EnableWCharSupport is set to 1.

2. The driver returns the VARCHAR types when the Hybrid Data Pipeline ODBC driver's connection option EnableWCharSupport is set to 1.
**Microsoft SQL Server data types**

The following table shows how the Microsoft SQL Server and Windows Azure SQL Database data types are mapped to the standard data types for ODBC and JDBC.

**Note:** When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, the varchar(max) type is mapped to the Unicode type SQL_WLONGVARCHAR.

<table>
<thead>
<tr>
<th>SQL Server data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bigint</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>binary</td>
<td>BINARY</td>
<td>SQL_BINARY(-2)</td>
</tr>
<tr>
<td>bit</td>
<td>BIT</td>
<td>SQL_BIT(-7)</td>
</tr>
<tr>
<td>char</td>
<td>CHAR</td>
<td>SQL_CHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>date</td>
<td>DATE</td>
<td>SQL_TYPE_DATE(91)</td>
</tr>
<tr>
<td>datetime</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>datetimeoffset</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>decimal</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>float</td>
<td>FLOAT</td>
<td>SQL_FLOAT(6)</td>
</tr>
<tr>
<td>int</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>image</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>money</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>nchar</td>
<td>CHAR</td>
<td>SQL_CHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>ntext</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
</tbody>
</table>

---

30 The connectivity service returns the WCHAR types when the Hybrid Data Pipeline ODBC driver's connection option EnableWCharSupport is set to 1.
31 The connectivity service returns the WCHAR types when the Hybrid Data Pipeline ODBC driver's connection option EnableWCharSupport is set to 1.
32 Supported only on Microsoft SQL Server 2008 and higher.
<table>
<thead>
<tr>
<th>SQL Server data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric</td>
<td>NUMERIC</td>
<td>SQL_NUMERIC(2)</td>
</tr>
<tr>
<td>nvarchar</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>nvarchar(max)</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>real</td>
<td>REAL</td>
<td>SQL_REAL(7)</td>
</tr>
<tr>
<td>smalldatetime</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>smallint</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT(5)</td>
</tr>
<tr>
<td>smallmoney</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>sql_variant</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>sysname</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>text</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>time</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>timestamp</td>
<td>BINARY</td>
<td>SQL_BINARY(-2)</td>
</tr>
<tr>
<td>tinyint</td>
<td>TINYINT</td>
<td>SQL_TINYINT(-6)</td>
</tr>
<tr>
<td>uniqueidentifier</td>
<td>CHAR</td>
<td>SQL_CHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>varbinary</td>
<td>VARBINARY</td>
<td>SQL_VARBINARY(-3)</td>
</tr>
<tr>
<td>varbinary(max)</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>varchar</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>varchar(max)</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>xml</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
</tbody>
</table>

*Time mapping changes based on the setting of the Fetch TWFS as Time option.*
### MySQL data types

The following table shows how the MySQL data types are mapped to the standard data types for ODBC and JDBC.

**Note:** When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, the varchar(max) type is mapped to the Unicode type SQL_WLONGVARCHAR.

<table>
<thead>
<tr>
<th>MySQL data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>BIGINT UNSIGNED</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>BINARY</td>
<td>BINARY</td>
<td>SQL_BINARY</td>
</tr>
<tr>
<td>BIT</td>
<td>BIT</td>
<td>SQL_BINARY</td>
</tr>
<tr>
<td>BLOB</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>CHAR</td>
<td>CHAR</td>
<td>SQL_CHAR or SQL_WCHAR</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
</tr>
<tr>
<td>DATETIME</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>DECIMAL UNSIGNED</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>DOUBLE UNSIGNED</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>FLOAT</td>
<td>REAL</td>
<td>SQL_REAL</td>
</tr>
<tr>
<td>FLOAT UNSIGNED</td>
<td>REAL</td>
<td>SQL_REAL</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>INTEGER UNSIGNED</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>LONGBLOB</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>LONGTEXT</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR or SQL_WLONGVARCHAR</td>
</tr>
<tr>
<td>MEDIUMBLOB</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
</tbody>
</table>
### Oracle data types

The following table shows how the Oracle data types are mapped to the standard data types for ODBC and JDBC.

**Note:** When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, NCHAR types are mapped to the Unicode types SQL_WCHAR, SQL_WVARCHAR, and SQL_WLONGVARCHAR.

#### Table 160: Oracle data types

<table>
<thead>
<tr>
<th>Oracle data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BFile</td>
<td>BLOB</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>Oracle data type</td>
<td>JDBC data type</td>
<td>ODBC data type</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Binary_Double</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>Binary_Float</td>
<td>REAL</td>
<td>SQL_REAL(7)</td>
</tr>
<tr>
<td>Blob</td>
<td>BLOB</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>Char</td>
<td>CHAR</td>
<td>SQL_WCHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>Clob</td>
<td>CLOB</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>Date</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>Long</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>Long Raw</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARCHAR(-4)</td>
</tr>
<tr>
<td>NChar</td>
<td>CHAR</td>
<td>SQL_WCHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>NClob</td>
<td>CLOB</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>Number</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>NVarchar2</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>Raw</td>
<td>VARBINARY</td>
<td>SQL_VARBINARY(-3)</td>
</tr>
<tr>
<td>Timestamp</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>Timestamp with Local Timezone</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>Timestamp with Timezone</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>UrowId</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>Varchar</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
</tbody>
</table>

34 Supported only on Oracle 10g and higher.
35 Supported only on Oracle 9i and higher.
36 Timestamp with timezone mapping changes based on the setting of the Fetch TSWTZ as Timestamp option only on Oracle 10g R2 and higher.
37 Timestamp with timezone mapping changes based on the setting of the Fetch TSWTZ as Timestamp option only on Oracle 10g R2 and higher.
Oracle Marketing Cloud (Eloqua) data types

The following table lists the Oracle Marketing Cloud data types and their equivalents for JDBC and ODBC.

Table 161: Oracle Marketing Cloud data types

<table>
<thead>
<tr>
<th>Oracle Marketing Cloud type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARRAY</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>BOOLEAN</td>
<td>SQL_BIT (-7)</td>
</tr>
<tr>
<td>DATETIME</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>DURATION</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>SQL_INTEGER (-4)</td>
</tr>
<tr>
<td>LARGETEXT</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>LONG</td>
<td>BIGINT</td>
<td>BIGINT (-5)</td>
</tr>
<tr>
<td>TEXT</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>URL</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
</tbody>
</table>

Note: Columns that are named as "ID" are mapped to BIGINT.

Oracle Sales Cloud data types

The following table lists the Oracle Sales Cloud data types and their equivalents for JDBC and ODBC.

---

38 Supported only on Oracle 9i R2 and higher.
**Oracle Sales Cloud data types**

The following table lists the Oracle Sales Cloud data types and their equivalents for JDBC and ODBC.

<table>
<thead>
<tr>
<th>Oracle Sales Cloud data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOLEAN</td>
<td>BOOLEAN</td>
<td>SQL_BIT(-7)</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>INTEGER&lt;sup&gt;39&lt;/sup&gt;</td>
<td>INTEGER or BIGINT or DECIMAL</td>
<td>SQL_INTEGER(4) or SQL_BIGINT(-5) or SQL_DECIMAL(3)</td>
</tr>
<tr>
<td>LONG</td>
<td>BIGINT</td>
<td>BIGINT(-5)</td>
</tr>
<tr>
<td>LONGSTRING</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR(-10) or SQL_LONGVARCHAR(-1)</td>
</tr>
<tr>
<td>STRING&lt;sup&gt;40&lt;/sup&gt;</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12)</td>
</tr>
</tbody>
</table>

**Oracle Service Cloud data types**

The following table lists the Oracle Service Cloud data types and their equivalents for JDBC and ODBC.

<table>
<thead>
<tr>
<th>Oracle Service Cloud data type</th>
<th>Documented Name</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BASE_64_BINARY</td>
<td>base64binary</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>boolean</td>
<td>BOOLEAN</td>
<td>SQL_BIT(-7)</td>
</tr>
<tr>
<td>DATE</td>
<td>date</td>
<td>DATE</td>
<td>SQL_TYPE_DATE(91)</td>
</tr>
<tr>
<td>DATETIME</td>
<td>datet ime</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>double</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>ID</td>
<td>ID</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>INTEGER</td>
<td>int</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>LONG</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
</tbody>
</table>

<sup>39</sup> When precision is less than or equal to 9, INTEGER is mapped as INTEGER for JDBC and SQL_INTEGER(4) for ODBC. When precision is greater than 9, INTEGER is mapped as BIGINT for JDBC and SQL_BIGINT(-5) for ODBC. When no precision is specified, INTEGER is mapped as DECIMAL for JDBC with a precision of 19 and a scale of 4. Similarly, for ODBC, when no precision is specified, INTEGER is mapped as SQL_DECIMAL(3) with a precision of 19 and a scale of 4.

<sup>40</sup> When no precision for STRING fields is offered in the metadata, STRING is mapped as VARCHAR with a length of 4000 characters for JDBC and SQL_VARCHAR(12) with a length of 4000 characters for ODBC. When precision for STRING columns is available, the precision is maintained and STRING is mapped as VARCHAR for JDBC and SQL_VARCHAR(12) for ODBC.
PostgreSQL data types

The following table shows how the PostgreSQL data types are mapped to the standard data types for ODBC and JDBC.

Note: When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, the varchar(max) type is mapped to the Unicode type SQL_WLONGVARCHAR.

Table 164: PostgreSQL data types

<table>
<thead>
<tr>
<th>PostgreSQL data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>BIGSERIAL</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>BIT</td>
<td>BIT or BINARY</td>
<td>SQL_BIT(-7) or SQL_BINARY(-2)</td>
</tr>
<tr>
<td>BIT VARYING</td>
<td>BINARY</td>
<td>SQL_BINARY(-2)</td>
</tr>
<tr>
<td>BOOLEAN</td>
<td>BOOLEAN</td>
<td>SQL_BIT(-7)</td>
</tr>
<tr>
<td>BYTEA</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(4)</td>
</tr>
<tr>
<td>CHARACTER</td>
<td>CHAR</td>
<td>SQL_WCHAR(-8) or SQL_CHAR(1)</td>
</tr>
<tr>
<td>CHARACTER VARYING</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR(-9) or SQL_VARCHAR(12)</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>SQL_TYPE_DATE(91)</td>
</tr>
<tr>
<td>DOUBLE PRECISION</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>INTEGER</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>NUMERIC</td>
<td>NUMERIC</td>
<td>SQL_NUMERIC(2)</td>
</tr>
<tr>
<td>REAL</td>
<td>REAL</td>
<td>SQL_REAL(7)</td>
</tr>
</tbody>
</table>
### Progress OpenEdge data types

The following table lists the Progress OpenEdge data types and their equivalents for JDBC and ODBC.

**Table 165: OpenEdge data types**

<table>
<thead>
<tr>
<th>OpenEdge data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit</td>
<td>BIT</td>
<td>SQL_BIT(-7)</td>
</tr>
<tr>
<td>tinyint</td>
<td>TINYINT</td>
<td>SQL_TINYINT(-6)</td>
</tr>
<tr>
<td>bigint</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>lvarchar</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR(-1) or SQL_WLONGVARCHAR(-10)</td>
</tr>
<tr>
<td>lvarbinary</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>blob</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>varbinary</td>
<td>VARBINARY</td>
<td>SQL_VARBINARY(-3)</td>
</tr>
<tr>
<td>binary</td>
<td>BINARY</td>
<td>SQL_BINARY(-2)</td>
</tr>
<tr>
<td>lvarchar</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR(-1) or SQL_WLONGVARCHAR(-10)</td>
</tr>
<tr>
<td>clob</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR(-1) or SQL_WLONGVARCHAR(-10)</td>
</tr>
<tr>
<td>character</td>
<td>CHAR</td>
<td>SQL_CHAR(1) or SQL WCHAR(-8)</td>
</tr>
<tr>
<td>OpenEdge data type</td>
<td>JDBC data type</td>
<td>ODBC data type</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>timestamp with timezone</td>
<td>CHAR</td>
<td>SQL_CHAR(1) or SQL_WCHAR(-8)</td>
</tr>
<tr>
<td>numeric</td>
<td>NUMERIC</td>
<td>SQL_NUMERIC(2)</td>
</tr>
<tr>
<td>integer</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>smallint</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT(5)</td>
</tr>
<tr>
<td>real</td>
<td>REAL</td>
<td>SQL_REAL(7)</td>
</tr>
<tr>
<td>double precision</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>float</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>varchar</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12) or SQL_WVARCHAR(-9)</td>
</tr>
<tr>
<td>date</td>
<td>DATE</td>
<td>SQL_TYPE_DATE(91)</td>
</tr>
<tr>
<td>time</td>
<td>TIME</td>
<td>SQL_TYPE_TIME(92)</td>
</tr>
<tr>
<td>timestamp</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
</tbody>
</table>

**Progress Rollbase data types**

In communication between your application and the data store, data is mapped several times to the type appropriate for sending and receiving components. The following Rollbase attribute types are supported for JDBC and ODBC.
<table>
<thead>
<tr>
<th>Data Store data type</th>
<th>Intermediary data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOLEAN</td>
<td>boolean</td>
<td>BOOLEAN</td>
<td>SQL_BIT(-7)</td>
</tr>
<tr>
<td>LONG</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>RELATIONSHIP</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>LONGREL</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>ORGDATA</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>STATUS</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>TEMPLATE</td>
<td>long</td>
<td>BIGINT</td>
<td>SQL_BIGINT(-5)</td>
</tr>
<tr>
<td>FILE</td>
<td>base64binary</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>IMAGE</td>
<td>base64binary</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY(-4)</td>
</tr>
<tr>
<td>TEXT</td>
<td>string</td>
<td>LONGVARCHAR</td>
<td>SQL_LONGVARCHAR(-1) or SQL_WLONGVARCHAR(-10)</td>
</tr>
<tr>
<td>INT</td>
<td>int</td>
<td>INTEGER</td>
<td>SQL_INTEGER(4)</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>double</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE(8)</td>
</tr>
<tr>
<td>STRING</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12) or SQL_WVARCHAR(-9)</td>
</tr>
<tr>
<td>ENCRYPTED</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12) or SQL_WVARCHAR(-9)</td>
</tr>
<tr>
<td>PICKLIST</td>
<td>picklist</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12) or SQL_WVARCHAR(-9)</td>
</tr>
<tr>
<td>LONGARR</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12) or SQL_WVARCHAR(-9)</td>
</tr>
<tr>
<td>PUBAPPFILE</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12) or SQL_WVARCHAR(-9)</td>
</tr>
<tr>
<td>AUTO</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_VARCHAR(12) or SQL_WVARCHAR(-9)</td>
</tr>
<tr>
<td>DATE</td>
<td>date</td>
<td>DATE</td>
<td>SQL_TYPE_DATE(91)</td>
</tr>
<tr>
<td>TIME</td>
<td>time</td>
<td>TIME</td>
<td>SQL_TYPE_TIME(92)</td>
</tr>
<tr>
<td>DATETIME</td>
<td>dateTime</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP(93)</td>
</tr>
</tbody>
</table>
Salesforce-type data types

Salesforce-type data stores include Salesforce, FinancialForce, Veeva CRM, and ServiceMax. In communication between your application and the data store, data is mapped several times to the type appropriate for the sending and receiving components. The following table lists the data types used by the data store, the JDBC or ODBC application, and the intermediary types.

Table 167: Supported data types

<table>
<thead>
<tr>
<th>Data Store data type</th>
<th>Intermediary data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANYTYPE</td>
<td>anytype</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR or SQL_VARCHAR</td>
</tr>
<tr>
<td>AUTONUMBER</td>
<td>string</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR or SQL_VARCHAR</td>
</tr>
<tr>
<td>BINARY</td>
<td>binary</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>CHECKBOX</td>
<td>boolean</td>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>COMBOBOX</td>
<td>combobox</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR or SQL_VARCHAR</td>
</tr>
<tr>
<td>DATACATEGORGROUPREFERENCE</td>
<td>DataCategoryGroupReference</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR or SQL_VARCHAR</td>
</tr>
<tr>
<td>EMAIL</td>
<td>email</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR or SQL_VARCHAR</td>
</tr>
<tr>
<td>ENCRYPTEDTEXT</td>
<td>encryptedtext</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR or SQL_VARCHAR</td>
</tr>
<tr>
<td>HTML</td>
<td>html</td>
<td>VARCHAR</td>
<td>SQL_WLONGVARCHAR or SQL_LONGVARCHAR</td>
</tr>
<tr>
<td>ID</td>
<td>id</td>
<td>LONGVARCHAR</td>
<td>SQL_WVARCHAR or SQL_VARCHAR</td>
</tr>
<tr>
<td>INT</td>
<td>double</td>
<td>INTEGER or DOUBLE</td>
<td>SQL_INTEGER or SQL_DOUBLE</td>
</tr>
</tbody>
</table>

41 You cannot create columns with this data type using the Create Table and AlterTable statements.

42 If scale = 0 and precision <= 9 and the NumberFieldMapping parameter under the Mapping tab is set to emulateInteger, this data type maps to SQL_INTEGER. If scale does not equal 0, precision > 9, or the NumberFieldMapping parameter under the Mapping tab is set to alwaysDouble, this data type maps to SQLDOUBLE.

43 If scale = 0 and precision <= 9 and the NumberFieldMapping parameter under the Mapping tab is set to emulateInteger, this data type maps to SQL_INTEGER. If scale does not equal 0, precision > 9, or the NumberFieldMapping parameter under the Mapping tab is set to alwaysDouble, this data type maps to SQLDOUBLE.
QUERYING AGAINST SALESFORCE EXTERNAL DATA SOURCES

Salesforce allows you to attach external data sources so they are exposed as if they are part of the Salesforce API. One of the mechanisms is OData, so if you have an OData data source, you can expose it through Salesforce via SOQL.

The following table provides the mapping from the underlying OData data types to the equivalent JDBC and ODBC data types. If you have connected tables to Salesforce using OData, you must use these data type mappings.

**Table 168: Supported data types**

<table>
<thead>
<tr>
<th>External OData data type</th>
<th>Salesforce data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edm.Binary</td>
<td>Not supported by Salesforce</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Edm.Boolean</td>
<td>CHECKBOX</td>
<td>BOOLEAN</td>
<td>SQL_BIT</td>
</tr>
</tbody>
</table>

---

For searchable columns, this data type maps to SQL_WVARCHAR or SQL_VARCHAR. For non-searchable columns, it maps to SQL_WLONGVARCHAR or SQL_LONGVARCHAR.

For searchable columns, this data type maps to VARCHAR. For non-searchable columns, it maps to LONGVARCHAR.
**Supported data types**

<table>
<thead>
<tr>
<th>External OData data type</th>
<th>Salesforce data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edm.Byte</td>
<td>NUMBER(3,0)</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>Edm.DateTime</td>
<td>DATE/TIME</td>
<td>DATETIME</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>Edm.DateTimeOffset</td>
<td>DATE/TIME</td>
<td>DATETIME</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>Edm.Decimal</td>
<td>NUMBER(14,4)</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>Edm.Double</td>
<td>NUMBER(10,8)</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>Edm.Guid</td>
<td>TEXT(64)</td>
<td>VARCHAR(64)</td>
<td>SQL_VARCHAR(64)</td>
</tr>
<tr>
<td>Edm.Int16</td>
<td>NUMBER(8,0)</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>Edm.Int32</td>
<td>NUMBER(18,0)</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>Edm.Int64</td>
<td>NUMBER(18,0)</td>
<td>DOUBLE</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>Edm.String</td>
<td>TEXT if the length is less than or equal to 255. Otherwise, LONGTEXTAREA</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR</td>
</tr>
<tr>
<td>TIME</td>
<td>Salesforce ignores fields of this type.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SugarCRM data types**

SugarCRM is implemented as a series of modules. When built, each module supports a set of data types. In addition, through the user interface, users can add tables that look and act like modules. Creating some fields triggers the creation of other fields that use different data types. For example, adding an "Address" adds extra columns for the components of an address, but does not create a column of type "Address".

Modules can also add their own custom data types. Data types that are not included in the following table are treated as strings (VARCHAR(255)). All data types, both those added from the user interface as well as those in the existing and user-created modules, are exposed through the SugarCRM metadata. Therefore, all are exposed as SQL tables.

The drop-down that the users select from has different names for some of these data types. Beginning with SugarCRM version 6.5, the set of supported data types changed. Existing modules may have references to data types that aren't visible from the user interface.
### Table 169: Supported data types

<table>
<thead>
<tr>
<th>Drop-down</th>
<th>Metaschema name</th>
<th>SQL type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>--</td>
<td>VARCHAR</td>
<td>Creates four more text fields for the address components. The field names are the name entered plus &quot;,_city&quot;, &quot;,_state&quot;, and &quot;,_country&quot;, as type &quot;varchar(100)&quot;, and &quot;,_postalcode&quot;, which is &quot;varchar(20)&quot;.</td>
</tr>
<tr>
<td>--</td>
<td>assigned_user_name</td>
<td>VARCHAR</td>
<td>Cannot be created via UI.</td>
</tr>
<tr>
<td>Checkbox</td>
<td>bool</td>
<td>BIT</td>
<td>Has three values: &quot;&quot;:&quot;&quot;, 1:&quot;Yes&quot;, 2:&quot;No&quot;. The default value is either checked or unchecked only.</td>
</tr>
<tr>
<td>Currency</td>
<td>currency</td>
<td>DECIMAL</td>
<td>The first time this is created in the record, a currency_id field of type currency_id is also created.</td>
</tr>
<tr>
<td>--</td>
<td>currency_id</td>
<td>VARCHAR</td>
<td>Created as a side-effect of creating the first currency column. It is always named &quot;currency_id&quot;.</td>
</tr>
<tr>
<td>Date</td>
<td>date</td>
<td>DATE</td>
<td>Default values include: yesterday, today, tomorrow, next week, next monday, next friday, two weeks, next month, first day of next month, three months, six months, next year.</td>
</tr>
<tr>
<td>--</td>
<td>datetime</td>
<td>TIMESTAMP</td>
<td>Cannot be created via UI.</td>
</tr>
<tr>
<td>Datetime</td>
<td>datetimecombo</td>
<td>TIMESTAMP</td>
<td>Defaults include those for date, and optional times. In addition for time, the hours 01-12:00,15,30,45:am/pm.</td>
</tr>
<tr>
<td>Decimal</td>
<td>decimal</td>
<td>DECIMAL</td>
<td></td>
</tr>
<tr>
<td>Dropdown</td>
<td>enum</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>email</td>
<td>VARCHAR</td>
<td>Cannot be created via UI.</td>
</tr>
<tr>
<td>Encrypt</td>
<td>encrypt</td>
<td>VARCHAR</td>
<td>Cannot be created via UI.</td>
</tr>
<tr>
<td>File</td>
<td>file</td>
<td>LONGVARCHAR</td>
<td></td>
</tr>
<tr>
<td>Float</td>
<td>float</td>
<td>FLOAT</td>
<td>Equivalent to Java Double.</td>
</tr>
<tr>
<td>Drop-down</td>
<td>Metaschema name</td>
<td>SQL type</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------</td>
<td>----------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>--</td>
<td>fullname</td>
<td>VARCHAR</td>
<td>This is a concatenation of the two name components first_name and last_name.</td>
</tr>
<tr>
<td>--</td>
<td>function</td>
<td>LONGVARCHAR</td>
<td>Cannot be created directly using the UI.</td>
</tr>
<tr>
<td>HTML</td>
<td>html</td>
<td>LONGVARCHAR</td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>id</td>
<td>LONGVARCHAR</td>
<td>Cannot be created directly using the UI.</td>
</tr>
<tr>
<td>IFrame</td>
<td>iframe</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>Image</td>
<td>image</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>Integer</td>
<td>int</td>
<td>INTEGER</td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>json</td>
<td>LONGVARCHAR</td>
<td>Cannot be created directly using the UI.</td>
</tr>
<tr>
<td>--</td>
<td>link</td>
<td>VARCHAR</td>
<td>Cannot be created directly using the UI.</td>
</tr>
<tr>
<td>--</td>
<td>long</td>
<td>BIGINT</td>
<td>Cannot be created directly using the UI.</td>
</tr>
<tr>
<td>--</td>
<td>longtext</td>
<td>LONGVARCHAR</td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>modified_user_name</td>
<td>VARCHAR</td>
<td>Cannot be created directly using the UI.</td>
</tr>
<tr>
<td>Multiselect</td>
<td>multienum</td>
<td>VARCHAR</td>
<td>Returned as comma-separated values.</td>
</tr>
<tr>
<td>--</td>
<td>name</td>
<td>VARCHAR</td>
<td>Cannot be created directly with the UI.</td>
</tr>
<tr>
<td>--</td>
<td>none</td>
<td></td>
<td>If the metadata returns a data type of &quot;none&quot;, the column is ignored.</td>
</tr>
<tr>
<td>Parent</td>
<td>parent</td>
<td>VARCHAR</td>
<td>Supports the SugarCRM &quot;Flex Relate&quot; feature, which allows the type of the link target to be set dynamically at runtime.</td>
</tr>
</tbody>
</table>
### Sybase data types

The following table shows how the Sybase data types are mapped to the standard data types for ODBC and JDBC.

<table>
<thead>
<tr>
<th>Drop-down</th>
<th>Metaschema name</th>
<th>SQL type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>parent_type</td>
<td>VARCHAR</td>
<td>Supports the SugarCRM &quot;Flex Relate&quot; feature, which allows the type of the link target to be set dynamically at runtime.</td>
</tr>
<tr>
<td>Password</td>
<td>password</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>PHONE</td>
<td>phone</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>Radio</td>
<td>radioenum</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>Relate</td>
<td>relate</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>short</td>
<td>INTEGER</td>
<td>Cannot be created using the UI.</td>
</tr>
<tr>
<td>--</td>
<td>string</td>
<td>VARCHAR</td>
<td>Cannot be created using the UI.</td>
</tr>
<tr>
<td>--</td>
<td>team_list</td>
<td>VARCHAR</td>
<td>Cannot be created using the UI.</td>
</tr>
<tr>
<td>Textarea</td>
<td>text</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>TextField</td>
<td>varchar</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>time</td>
<td>TIME</td>
<td></td>
</tr>
<tr>
<td>TimePeriod</td>
<td>timeperiod</td>
<td>VARCHAR</td>
<td>Cannot be created directly with the UI.</td>
</tr>
<tr>
<td>URL</td>
<td>url</td>
<td>VARCHAR</td>
<td></td>
</tr>
<tr>
<td>--</td>
<td>user_name</td>
<td>VARCHAR</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** When the EnableWCharSupport connection parameter is set to true for the Hybrid Data Pipeline Driver for ODBC, character types are mapped to the corresponding ODBC W-Types. For example, the varchar(max) type is mapped to the Unicode type SQL_WLONGVARCHAR.
### Table 170: Sybase data types

<table>
<thead>
<tr>
<th>Sybase data type</th>
<th>JDBC data type</th>
<th>ODBC data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGDATETIME</td>
<td>TIMESTAMP</td>
<td>SQL_DATETIME</td>
</tr>
<tr>
<td>BIGINT</td>
<td>BIGINT</td>
<td>SQL_BIGINT</td>
</tr>
<tr>
<td>BIGTIME</td>
<td>TIME or TIMESTAMP</td>
<td>SQL_DATETIME</td>
</tr>
<tr>
<td>BINARY</td>
<td>BINARY</td>
<td>SQL_BINARY</td>
</tr>
<tr>
<td>BIT</td>
<td>BIT</td>
<td>SQL_BIT</td>
</tr>
<tr>
<td>CHAR</td>
<td>CHAR</td>
<td>SQL_CHAR or SQL_WCHAR</td>
</tr>
<tr>
<td>DATE</td>
<td>DATE</td>
<td>SQL_TYPE_DATE</td>
</tr>
<tr>
<td>DATETIME</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>DECIMAL</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>FLOAT</td>
<td>FLOAT</td>
<td>SQL_FLOAT</td>
</tr>
<tr>
<td>IMAGE</td>
<td>LONGVARBINARY</td>
<td>SQL_LONGVARBINARY</td>
</tr>
<tr>
<td>INT</td>
<td>INTEGER</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>MONEY</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>NUMERIC</td>
<td>NUMERIC</td>
<td>SQL_NUMERIC</td>
</tr>
<tr>
<td>REAL</td>
<td>REAL</td>
<td>SQL_REAL</td>
</tr>
<tr>
<td>SMALLDATETIME</td>
<td>TIMESTAMP</td>
<td>SQL_TYPE_TIMESTAMP</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>SMALLINT</td>
<td>SQL_SMALLINT</td>
</tr>
<tr>
<td>SMALLMONEY</td>
<td>DECIMAL</td>
<td>SQL_DECIMAL</td>
</tr>
<tr>
<td>SYSNAME</td>
<td>VARCHAR</td>
<td>SQL_WVARCHAR or SQLVARCHAR</td>
</tr>
<tr>
<td>TEXT</td>
<td>LONGVARCHAR</td>
<td>SQL_WLONGVARCHAR or SQL_LONGVARCHAR</td>
</tr>
<tr>
<td>TIME</td>
<td>TIME or TIMESTAMP</td>
<td>SQL_TYPE_TIME</td>
</tr>
</tbody>
</table>

---

46 Supported only for Sybase 15.5 and higher.

47 Supported only for Sybase 15.0 and higher.

48 When `FetchTWFSasTime=true`, this Sybase data type is mapped to the JDBC TIME data type. When `FetchTWFSasTime=false`, this Sybase data type is mapped to the JDBC TIMESTAMMP data type.

49 Supported only for Sybase 12.5 and higher.

50 Time mapping changes based on the setting of the Fetch TWFS as Time option.
### Supported scalar functions

Support for scalar functions differs depending on the data store you are accessing. Each scalar function returns a single value based on the input value. The SQLGetInfo function returns information about supported functions.

Applications can construct SQL statements using the following syntax, where `scalar-function` is one of the functions listed in the topic for your data store.

```
{fn scalar-function}
```

For example:

```sql
SELECT {fn UCASE(NAME)} FROM EMP
```

### Scalar Function Support for Amazon Redshift

The table identifies the scalar functions that Hybrid Data Pipeline supports for Amazon Redshift.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

#### Table 171: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>DBNAME</td>
</tr>
</tbody>
</table>

51 When JDBCBehavior=0, the data type depends on the JVM the application uses. For JVMs earlier than Java SE 6, the first value applies. For Java SE 6 and higher, the second value applies.
<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURRENT_DATE</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>CURRENT_TIME</td>
<td>USERNAME</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>CURTIME</td>
<td></td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>FLOOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>LOG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POSITION</td>
<td>LOG10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>MOD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>PI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TRUNCATE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Scalar Function Support for Apache Hive**

The table identifies the scalar functions that Hybrid Data Pipeline supports for Apache Hive.
Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

**Table 172: Scalar Functions**

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>INSERT</td>
<td>ASIN</td>
<td><em>DAYOFMONTH</em></td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>ATAN</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>CEILING</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>COS</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>COT</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>DEGREES</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>EXP</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>FLOOR</td>
<td>TIMESTAMPAADD</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>LOG</td>
<td>TIMESTAMPDiff</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>LOG10</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>MOD</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>PI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAND</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROUND</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIN</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TAN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for Autonomous REST Connector

The table identifies the scalar functions that Hybrid Data Pipeline supports for REST services. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 173: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURRENT_DATE</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>CURRENT_TIME</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>CURRENT_TIMESTAMP</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>CURTIME</td>
<td></td>
</tr>
<tr>
<td>CONCAT</td>
<td>BITAND</td>
<td>DATEDIFF</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITOR</td>
<td>DATE_ADD</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITXOR</td>
<td>DATE_SUB</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>CEILING</td>
<td>DAY</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>DAYNAME</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>FLOOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>LOG</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>MOD</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>RAWTOHEX</td>
<td>PI</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>POWER</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>RADIANS</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>RAND</td>
<td>SECOND</td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for DB2

The table identifies the scalar functions that Hybrid Data Pipeline supports for DB2.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 174: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>COALESCE</td>
</tr>
<tr>
<td>BLOB</td>
<td>ABSVAL</td>
<td>CURTIME</td>
<td>DEREF</td>
</tr>
<tr>
<td>CHAR</td>
<td>ACOS</td>
<td>DATE</td>
<td>DLCOMMENT</td>
</tr>
<tr>
<td>CHR</td>
<td>ASIN</td>
<td>DAY</td>
<td>DLLINKTYPE</td>
</tr>
<tr>
<td>CLOB</td>
<td>ATAN</td>
<td>DAYNAME</td>
<td>DLURLCOMPLETE</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ATANH</td>
<td>DAYOFWEEK</td>
<td>DLURLPATH</td>
</tr>
<tr>
<td>DBCLOB</td>
<td>ATAN2</td>
<td>DAYOFYEAR</td>
<td>DLURLPATHONLY</td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BIGINT</td>
<td>DAYS</td>
<td>DLURLSCHEME</td>
</tr>
<tr>
<td>GRAPHIC</td>
<td>CEIL</td>
<td>HOUR</td>
<td>DLURLSERVER</td>
</tr>
<tr>
<td>HEX</td>
<td>CEILING</td>
<td>JULIAN_DAY</td>
<td>DLVALUE</td>
</tr>
<tr>
<td>INSERT</td>
<td>COS</td>
<td>MICROSECOND</td>
<td>EVENT_MON_STATE</td>
</tr>
<tr>
<td>LCASE</td>
<td>COSH</td>
<td>MIDNIGHT_SECONDS</td>
<td>GENERATE_UNIQUE</td>
</tr>
</tbody>
</table>

Scalar Functions Timedate Functions Numeric Functions String Functions

RTRIM ROUND SECONDS_SINCE_MIDNIGHT
SOUNDEX ROUNDMAGIC TIMESTAMPADD
SPACE SIGN TIMESTAMPDIFF
SUBSTR SIN TO_CHAR
SUBSTRING SQRT WEEK
UCASE TAN YEAR
UPPER TRUNCATE
<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>MINUTE</td>
<td>NODENUMBER</td>
</tr>
<tr>
<td>LENGTH</td>
<td>DECIMAL</td>
<td>MONTH</td>
<td>NULLIF</td>
</tr>
<tr>
<td>LOCATE</td>
<td>DEGREES</td>
<td>MONTHNAME</td>
<td>PARTITION</td>
</tr>
<tr>
<td>LONGVARCHAR</td>
<td>DIGITS</td>
<td>NOW</td>
<td>RAISE_ERROR</td>
</tr>
<tr>
<td>LONGVARCHAR</td>
<td>DOUBLE</td>
<td>QUARTER</td>
<td>TABLE_NAME</td>
</tr>
<tr>
<td>LOWER</td>
<td>EXP</td>
<td>SECOND</td>
<td>TABLE_SCHEMA</td>
</tr>
<tr>
<td>LTRIM</td>
<td>FLOAT</td>
<td>TIME</td>
<td>TRANSLATE</td>
</tr>
<tr>
<td>POSSTR</td>
<td>FLOOR</td>
<td>TIMESTAMP</td>
<td>TYPE_ID</td>
</tr>
<tr>
<td>REPEAT</td>
<td>INTEGER</td>
<td>TIMESTAMP_ISO</td>
<td>TYPE_NAME</td>
</tr>
<tr>
<td>REPLACE</td>
<td>LN</td>
<td>TIMESTAMPDIFF</td>
<td>TYPE_SCHEMA</td>
</tr>
<tr>
<td>RIGHT</td>
<td>LOG</td>
<td>WEEK</td>
<td>VALUE</td>
</tr>
<tr>
<td>RTRIM</td>
<td>LOG10</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>MOD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTR</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRUNC</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRUNCATE</td>
<td>REAL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPPER</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARCHAR</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VARGRAPHIC</td>
<td>SINH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMALLINT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQRT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TAN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Scalar Function Support for Google Analytics

The table identifies the scalar functions supported by Hybrid Data Pipeline for Google Analytics. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

**Table 175: Scalar Functions**

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAY</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>DAYNAME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITAND</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITOR</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>BITXOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>FLOOR</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>LOG</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>TO_CHAR</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>MOD</td>
<td>WEEK</td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for Greenplum

The table identifies the scalar functions that Hybrid Data Pipeline supports for Greenplum. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 176: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CHAR</td>
<td>ACOS</td>
<td>CURRENT_DATE</td>
<td>USER</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ASIN</td>
<td>CURRENT_TIME</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>ATAN</td>
<td>CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>ATAN2</td>
<td>CURTIME</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>CEILING</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>COS</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>COT</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for Informix

The table identifies the scalar functions that Hybrid Data Pipeline supports for Informix.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 177: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONCAT</td>
<td>ABS</td>
<td>CURDATE</td>
<td>DATABASE</td>
</tr>
<tr>
<td>LEFT</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>LTRIM</td>
<td>ASIN</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for Microsoft Dynamics

The table identifies the scalar functions that Hybrid Data Pipeline supports for Microsoft Dynamics. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 178: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAY</td>
<td>USER</td>
</tr>
</tbody>
</table>
## String Functions
- CHARACTER_LENGTH
- CONCAT
- DIFFERENCE
- HEXTORAW
- INSERT
- LCASE
- LEFT
- LENGTH
- LOCATE
- LOCATE_2
- LOWER
- LTRIM
- OCTET_LENGTH
- RAWTOHEX
- REPEAT
- REPLACE
- RIGHT
- RTRIM
- SOUNDEX
- SPACE
- SUBSTR
- SUBSTRING
- UCASE
- UPPER

## Numeric Functions
- ATAN2
- CEILING
- BITAND
- BITOR
- BITXOR
- COS
- COT
- DEGREES
- EXP
- FLOOR
- LOG
- LOG10
- MOD
- PI
- POWER
- RADIANS
- RAND
- ROUND
- SIGN
- SIN
- SQRT
- TAN
- TRUNCATE
- UCASE
- UPPER

## Timedate Functions
- DAYNAME
- DAYOFMONTH
- DAYOFWEEK
- DAYOFYEAR
- EXTRACT
- HOUR
- MINUTE
- MONTH
- MONTHNAME
- NOW
- SECOND
- TO_CHAR
- WEEK

## System Functions
- IFNULL
- RAWTOHEX
Scalar Function Support for Microsoft SQL Server

The table identifies the scalar functions that Hybrid Data Pipelines supports for Microsoft SQL Server. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 179: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ASIN</td>
<td>DAYNAME</td>
<td>USER</td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>ATAN</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>ATAN2</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>CEILING</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COS</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>COT</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>DEGREES</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>EXP</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>FLOOR</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>LOG</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>LOG10</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>MOD</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>PI</td>
<td>TIMESTAMPADD</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>POWER</td>
<td>TIMESTAMPDIFF</td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>RADIANS</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>RAND</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>ROUND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIGN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for MySQL

The table identifies the scalar functions that Hybrid Data Pipeline supports for MySQL. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 180: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ACOS</td>
<td>CURRENT_DATE</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ASIN</td>
<td>CURRENT_TIME</td>
<td>USER</td>
</tr>
<tr>
<td>INSERT</td>
<td>ATAN</td>
<td>CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>ATAN2</td>
<td>CURTIME</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>CEILING</td>
<td>DAYNAME</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>COS</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>COT</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>DEGREES</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>EXP</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>FLOOR</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>LOG</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>LOG10</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>MOD</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>PI</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>POWER</td>
<td>QUARTER</td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for Oracle

The table identifies the scalar functions that Hybrid Data Pipeline supports for Oracle.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 181: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>IFNULL</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURRENT_DATE</td>
<td>USER</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>CONCAT</td>
<td>ATAN</td>
<td>DAYNAME</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>ATAN2</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>CEILING</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COS</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>COT</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LOCATE2</td>
<td>FLOOR</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG</td>
<td>MONTHNAME</td>
<td></td>
</tr>
</tbody>
</table>
## Scalar Function Support for Oracle Marketing Cloud (Eloqua)

The table identifies the scalar functions that are supported for Oracle Marketing Cloud.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

### Table 182: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCTET_LENGTH</td>
<td>LOG10</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>MOD</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>PI</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>POWER</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>ROUND</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ASCII</th>
<th>ABS</th>
<th>CURDATE</th>
<th>CURSESSIONID</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAY</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>DAYNAME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITAND</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITOR</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>BITXOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>HOUR</td>
<td></td>
</tr>
</tbody>
</table>
### Scalar Function Support for Oracle Sales Cloud

The table identifies the scalar functions that Hybrid Data Pipeline supports for Oracle Sales Cloud. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

#### Table 183: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
</tbody>
</table>

---

Supported scalar functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>COALESCE</td>
<td>COALESCE</td>
<td>MINUTE</td>
<td>MINUTE</td>
</tr>
<tr>
<td>DATE_CUTOFF</td>
<td>DATE_CUTOFF</td>
<td>MONTH</td>
<td>MONTH</td>
</tr>
<tr>
<td>EXP</td>
<td>EXP</td>
<td>MONTHNAME</td>
<td>MONTHNAME</td>
</tr>
<tr>
<td>FLOOR</td>
<td>FLOOR</td>
<td>NOW</td>
<td>NOW</td>
</tr>
<tr>
<td>LOG</td>
<td>LOG</td>
<td>SECOND</td>
<td>SECOND</td>
</tr>
<tr>
<td>LOG10</td>
<td>LOG10</td>
<td>TO_CHAR</td>
<td>TO_CHAR</td>
</tr>
<tr>
<td>MOD</td>
<td>MOD</td>
<td>WEEK</td>
<td>WEEK</td>
</tr>
<tr>
<td>PI</td>
<td>PI</td>
<td>YEAR</td>
<td>YEAR</td>
</tr>
<tr>
<td>POWER</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RADIANS</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAND</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROUND</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIGN</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIN</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQRT</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TAN</td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRUNCATE</td>
<td>TRUNCATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>UCASE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPPER</td>
<td>UPPER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>OCTET_LENGTH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>String Functions</td>
<td>Numeric Functions</td>
<td>Timedate Functions</td>
<td>System Functions</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------</td>
<td>-------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAY</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>DAYNAME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITAND</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITOR</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>BITXOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>FLOOR</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>LOG</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>TO_CHAR</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>MOD</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>RAWTOHEX</td>
<td>PI</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTR</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Scalar Function Support for Oracle Service Cloud

The table identifies the scalar functions that Hybrid Data Pipeline supports for Oracle Service Cloud.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

**Table 184: Scalar Functions**

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAY</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>DAYNAME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITAND</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITOR</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>BITXOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>FLOOR</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>LOG</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>TO_CHAR</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>MOD</td>
<td>WEEK</td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for PostgreSQL

The table identifies the scalar functions that Hybrid Data Pipeline supports for PostgreSQL.
Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

**Table 185: Scalar Functions**

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAWTOHEX</td>
<td>PI</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTR</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>TRUNCATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPPER</td>
<td>ROUNDMAGIC</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CHAR</td>
<td>ACOS</td>
<td>CURRENT_DATE</td>
<td>USER</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ASIN</td>
<td>CURRENT_TIME</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>ATAN</td>
<td>CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>ATAN2</td>
<td>CURTIME</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>CEILING</td>
<td>DAYNAME</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>COS</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>COT</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
</tbody>
</table>
### Scalar Function Support for Progress OpenEdge

Applications connecting through JDBC or ODBC to Progress OpenEdge® can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

**Table 186: Scalar Functions for Progress OpenEdge**

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURRENT_DATE</td>
<td>DATABASE</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURRENT_TIME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>CURRENT_TIMESTAMP</td>
<td>ROWID</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>CURDATE</td>
<td>USER</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCATE_2</td>
<td>DEGREES</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>EXP</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>FLOOR</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>LOG</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>LOG10</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>MOD</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>PI</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>POWER</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>RADIANS</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RAND</td>
<td>TIMESTAMPADD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ROUND</td>
<td>TIMESTAMPDIFF</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SIGN</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SIN</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TRUNCATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>String Functions</td>
<td>Numeric Functions</td>
<td>Timedate Functions</td>
<td>System Functions</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------</td>
<td>-------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>CURTIME</td>
<td></td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>DAYNAME</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>COS</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>COT</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>DEGREES</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>EXP</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>FLOOR</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>LOG</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>MONTNAME</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>MOD</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>POSITION</td>
<td>PI</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>POWER</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>RADIANS</td>
<td>TIMESTAMPADD</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>RAND</td>
<td>TIMESTAMPDIFF</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>ROUND</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>SIGN</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TAN</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TRUNCATE</td>
</tr>
</tbody>
</table>

**Scalar Function Support for Progress Rollbase**

The table identifies the scalar functions that Hybrid Data Pipeline supports for Progress Rollbase.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.
### Table 187: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAY</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>DAYNAME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITAND</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITOR</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>BITXOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>FLOOR</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>LOG</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>TO_CHAR</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>MOD</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>RAWTOHEX</td>
<td>PI</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTR</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scalar Function Support for Salesforce-based data stores

The table identifies the scalar functions that Hybrid Data Pipeline supports for Salesforce-based data stores, including Force.com, ServiceMax, FinancialForce, and Veeva CRM.

Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

Table 188: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURRENT_DATE</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>CURRENT_TIME</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>CURRENT_TIMESTAMP</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>CURTIME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>BITAND</td>
<td>DATEDIFF</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITOR</td>
<td>DATE_ADD</td>
<td></td>
</tr>
<tr>
<td>HEXTORAW</td>
<td>BITXOR</td>
<td>DATE_SUB</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>CEILING</td>
<td>DAY</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>DAYNAME</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>FLOOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>LOG</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>MINUTE</td>
<td></td>
</tr>
</tbody>
</table>
### Scalar Function Support for SugarCRM

The table identifies the scalar functions supported by Hybrid Data Pipeline for SugarCRM. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

**Table 189: Scalar Functions**

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>CURDATE</td>
<td>CURSESSIONID</td>
</tr>
<tr>
<td>BIT_LENGTH</td>
<td>ACOS</td>
<td>CURTIME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ASIN</td>
<td>DATEDIFF</td>
<td>IDENTITY</td>
</tr>
<tr>
<td>CHAR_LENGTH</td>
<td>ATAN</td>
<td>DAY</td>
<td>USER</td>
</tr>
<tr>
<td>CHARACTER_LENGTH</td>
<td>ATAN2</td>
<td>DAYNAME</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>CEILING</td>
<td>DAYOFMONTH</td>
<td></td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>BITAND</td>
<td>DAYOFWEEK</td>
<td></td>
</tr>
</tbody>
</table>
### Scalar Function Support for Sybase

The table identifies the scalar functions that Hybrid Data Pipeline supports for Sybase. Applications connecting through JDBC or ODBC can use the following scalar functions in expressions. For syntax details, consult your JDBC or ODBC documentation.

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEXTORAW</td>
<td>BITOR</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>BITXOR</td>
<td>EXTRACT</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>COS</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COT</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>DEGREES</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>EXP</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>LOCATE_2</td>
<td>FLOOR</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LOWER</td>
<td>LOG</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>LOG10</td>
<td>TO_CHAR</td>
<td></td>
</tr>
<tr>
<td>OCTET_LENGTH</td>
<td>MOD</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>RAWTOHEX</td>
<td>PI</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPLACE</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>RAND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>ROUND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>SIGN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>SIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTR</td>
<td>SQRT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>TAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>TRUNCATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPPER</td>
<td>ROUNDMAGIC</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 190: Scalar Functions

<table>
<thead>
<tr>
<th>String Functions</th>
<th>Numeric Functions</th>
<th>Timedate Functions</th>
<th>System Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ABS</td>
<td>DAYNAME</td>
<td>DATABASE</td>
</tr>
<tr>
<td>CHAR</td>
<td>ACOS</td>
<td>DAYOFMONTH</td>
<td>IFNULL</td>
</tr>
<tr>
<td>CONCAT</td>
<td>ASIN</td>
<td>DAYOFWEEK</td>
<td>USER</td>
</tr>
<tr>
<td>DIFFERENCE</td>
<td>ATAN</td>
<td>DAYOFYEAR</td>
<td></td>
</tr>
<tr>
<td>INSERT</td>
<td>ATAN2</td>
<td>HOUR</td>
<td></td>
</tr>
<tr>
<td>LCASE</td>
<td>CEILING</td>
<td>MINUTE</td>
<td></td>
</tr>
<tr>
<td>LEFT</td>
<td>COS</td>
<td>MONTH</td>
<td></td>
</tr>
<tr>
<td>LENGTH</td>
<td>COT</td>
<td>MONTHNAME</td>
<td></td>
</tr>
<tr>
<td>LOCATE</td>
<td>DEGREES</td>
<td>NOW</td>
<td></td>
</tr>
<tr>
<td>LTRIM</td>
<td>EXP</td>
<td>QUARTER</td>
<td></td>
</tr>
<tr>
<td>REPEAT</td>
<td>FLOOR</td>
<td>SECOND</td>
<td></td>
</tr>
<tr>
<td>RIGHT</td>
<td>LOG</td>
<td>TIMESTAMPADD</td>
<td></td>
</tr>
<tr>
<td>RTRIM</td>
<td>LOG10</td>
<td>TIMESTAMPDIFF</td>
<td></td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>MOD</td>
<td>WEEK</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td>PI</td>
<td>YEAR</td>
<td></td>
</tr>
<tr>
<td>SUBSTRING</td>
<td>POWER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCASE</td>
<td>RADIANS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROUND</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIGN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQRT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TAN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRUNCATE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Using Salesforce reports

The Salesforce-type data stores provide reporting functionality. Hybrid Data Pipeline exposes custom reports defined in a data store as stored procedures. An application can obtain a list of the reports by calling the SQLProcedures catalog function. The names of the reports that can be invoked through the Hybrid Data Pipeline connectivity service are listed in the PROCEDURE_NAME name column of the SQLProcedures results.

Note that if you are using a standard report, you must save it as a custom report using the tabular, summary, or matrix format. Check with your Salesforce administrator to make sure that you have the necessary permissions to create custom reports.

Salesforce data store reports

Salesforce-based data stores deliver several types of standard reports that users can customize. The connectivity service can access custom reports that use the tabular, summary, or matrix formats. If you want to access a standard report, you can save most standard reports as custom reports and access them through the Hybrid Data Pipeline connectivity service. Check with your Salesforce administrator to make sure that you have the necessary permissions to create custom reports.

Salesforce-based data stores organize reports into folders. The connectivity service incorporates the folder name and report name into the procedure name reported by SQLProcedures. The name is created by prepending the folder name to the report name using an underscore to join them. Additionally, any spaces in the report or folder names are replaced with an underscore character. Like all identifier name metadata returned by the connectivity service, the procedure name is upper case. For example, if a report named Opportunity Pipeline is in the folder Opportunity Reports, it would be rendered as:

OPPORTUNITY_REPORTS_OPPORTUNITY_PIPELINE

An application invokes a report using the standard Call escape syntax, {call report name}, and the appropriate mechanisms for handling a resultset.

The following example shows one way to invoke the Opportunity Pipeline report using the driver for ODBC:

```java
SQLRETURN  retVal;
HSTMT  hStmt = NULL;
SQLNCHAR* sql;
sql = L"{call OPPORTUNITY_REPORTS_OPPORTUNITY_PIPELINE}";
retVal = SQLExecDirect(hStmt, sql, SQL_NTS);
if (SQL_SUCCESS == retVal) {
   // process results
}
```

The following example shows one way to invoke the Opportunity Pipeline report using the driver for JDBC:

```java
String sql = "{call OPPORTUNITY_REPORTS_OPPORTUNITY_PIPELINE()}";
CallableStatement callStmt = con.prepareCall(sql);
boolean isResultSet = callStmt.execute();
if (isResultSet) {
   resultSet = callStmt.getResultSet();
   // process the resultset
}
```

Note: Reports in the joined, or multi-block, format are not supported.
**Supported SQL and Extensions**

Hybrid Data Pipeline supports the SQL statements and extensions described in this section.

The SQL statements supported are similar in many cases for any data store. However, in some cases, the data store has different levels of SQL support.

<table>
<thead>
<tr>
<th>Data store</th>
<th>Supported SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Redshift</td>
<td>Hybrid Data Pipeline supports the SQL supported by Amazon Redshift. Refer to your Amazon Redshift documentation for details on SQL syntax.</td>
</tr>
</tbody>
</table>
| Autonomous REST Connector   | • Alter Session (EXT) on page 970  
                              | • Select on page 981                                                                                                                         |
| DB2                         | Hybrid Data Pipeline supports the SQL supported by DB2. Refer to your DB2 documentation for details on SQL syntax.                             |
| Microsoft Dynamics CRM Online | • Alter Session (EXT) on page 970  
                                  | • Delete on page 978                                                                  |
| Google Analytics            | • Alter Session (EXT) on page 970  
                              | • Explain Plan on page 979                                                           |
| Greenplum                   | • Alter Session (EXT) on page 970  
                              | • Select on page 981                                                                                                                       |
| Informix                    | Hybrid Data Pipeline supports the SQL supported by Greenplum. Refer to your Greenplum documentation for details on SQL syntax.             |
|                             | Hybrid Data Pipeline supports the SQL supported by Informix. Refer to your Informix documentation for details on SQL syntax.              |
## Chapter 8: Querying data stores with SQL

<table>
<thead>
<tr>
<th>Data store</th>
<th>Supported SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Microsoft SQL Server</strong></td>
<td>Hybrid Data Pipeline supports the SQL used by Microsoft SQL Server. Refer to the Microsoft SQL Server documentation for details on the SQL syntax.</td>
</tr>
<tr>
<td><strong>Oracle</strong></td>
<td>Hybrid Data Pipeline supports the SQL supported by Oracle. Refer to your Oracle documentation for details on SQL syntax.</td>
</tr>
</tbody>
</table>
| **Oracle Marketing Cloud**    | • Alter Session (EXT) on page 970  
                                | • Delete on page 978                                                                                                                        |
| (Eloqua)                      | • Explain Plan on page 979                                                           
                                | • Insert on page 979                                                                                                                        |
|                               | • Select on page 981                                                                                                                         |
|                               | • Update on page 991                                                                                                                         |
| **Oracle Sales Cloud**        | • Alter Session (EXT) on page 970  
                                | • Explain Plan on page 979                                                                                                                   |
|                               | • Select on page 981                                                                                                                         |
| **Oracle Service Cloud**      | • Alter Session (EXT) on page 970  
                                | • Delete on page 978                                                                                                                        |
|                               | • Explain Plan on page 979                                                           
<pre><code>                            | • Insert on page 979                                                                                                                         |
</code></pre>
<p>|                               | • Select on page 981                                                                                                                         |
|                               | • Update on page 991                                                                                                                         |
| <strong>PostgreSQL</strong>                | Hybrid Data Pipeline supports the SQL supported by PostgreSQL. Refer to your PostgreSQL documentation for details on SQL syntax.                |</p>
<table>
<thead>
<tr>
<th>Data store</th>
<th>Supported SQL</th>
</tr>
</thead>
</table>
| Progress OpenEdge                              | Hybrid Data Pipeline supports the SQL supported by the Progress OpenEdge Database, with the following exceptions:  
  • Stored Procedure Output Parameters  
  • Multiple Results  
  • COMMIT, ROLLBACK, and SET TRANSACTION ISOLATION LEVEL 52  
  Refer to OpenEdge SQL Statements in the OpenEdge documentation for details. |
| Progress Rollbase                              | • Alter Session (EXT) on page 970  
  • Delete on page 978  
  • Explain Plan on page 979  
  • Insert on page 979  
  • Select on page 981  
  • Update on page 991 |
| Salesforce-based data stores (Salesforce, Force.com, FinancialForce, ServiceMax, and Veeva CRM) | • Alter Session (EXT) on page 970  
  • Alter Table for Salesforce on page 971  
  • Create Table for Salesforce on page 974  
  • Delete on page 978  
  • Drop Table for Salesforce on page 978  
  • Explain Plan on page 979  
  • Insert on page 979  
  • Select on page 981  
  • Update on page 991 |
| SugarCRM                                       | • Alter Session (EXT) on page 970  
  • Explain Plan on page 979  
  • Select on page 981 |
| Sybase                                         | Hybrid Data Pipeline supports the SQL supported by Sybase. Refer to your Sybase documentation for details on SQL syntax. |

52 When using Hybrid Data Pipeline to access OpenEdge data, you (or your applications) do not explicitly control transactions. Instead, all SQL statements are auto-committed.
Alter Session (EXT)

Purpose
The Alter Session statement allows you to change various attributes of a connection session.

Syntax

ALTER SESSION SET attribute_name=value

where:

attribute_name

Specifies the name of the attribute to be changed.

table

Refers to the specific value setting for that attribute.

The following table lists session attributes and describes them.

Table 192: Alter Session Attributes

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Session Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current_Schema</td>
<td>Database</td>
<td>Sets the current schema for the database session. The current schema is the schema used when an identifier in a SQL statement is unqualified. The string value must be the name of a schema visible in the session. For example: ALTER SESSION SET CURRENT_SCHEMA=sforce</td>
</tr>
</tbody>
</table>
### Attribute Name | Session Type | Description
---|---|---
Stmt_Call_Limit | Database | Sets the maximum number of Web service calls the driver can make in executing a statement. Setting the `Stmt_Call_Limit` attribute has the same effect as setting the `StmtCallLimit` connection option. It sets the default Web service call limit used by any statement on the connection. Executing this command on a statement overrides the previously set `StmtCallLimit` for the connection. The value specified must be a positive integer or 0. The value 0 means that no call limit exists. For example:

```
ALTER SESSION SET STMT_CALL_LIMIT=10
```

Ws_Call_Count | Remote | Resets the Web service call count of a session to the value specified. The value must be zero or a positive integer. `Ws_Call_Count` represents the total number of Web service calls made to the data store instance for the current session. For example:

```
ALTER SESSION SET sforce.WS_CALL_COUNT=0
```

The current value of `Ws_Call_Count` can be obtained by referring to the `System_Remote_Sessions` system table. For example:

```
SELECT * FROM information_schema.system_remote_sessions
WHERE session_id = cursessionid()
```

---

### Alter Table for Salesforce

**Purpose**

The Alter Table statement adds a column, removes a column, or redefines a column in a table.

**Syntax**

```
ALTER TABLE table_name
[add_clause]
[drop_clause]
```

where:

- **table_name**
  - specifies an existing table.

- **add_clause**

  specifies a column or a foreign key constraint to be added to the table. See Add Clause: Columns on page 972 and Add Clause: Constraints on page 973

- **drop_clause**

  specifies a column to be dropped from the table. See Drop Clause: Columns on page 973 for a complete explanation.
Notes

- You cannot drop a constraint from a table.

Add Clause: Columns

Purpose
Supported only for Salesforce-based data stores. Adds a column to an existing table. This clause is optional.

Syntax

```
ADD [COLUMN] column_name Datatype ...  
[DEFAULT default_value] [NOTNULL] [EXT_ID] [PRIMARY KEY]  
[START WITH starting_value]
```

where:

- `default_value`
  
  specifies the default value to be assigned to the column. See Column Definition on page 974 for details.

- `starting_value`
  
  specifies the starting value for the Identity column. The default start value is 0.

Notes

- If `NOT NULL` is specified and the table is not empty, a default value must be specified. In all other respects, this command is the equivalent of a column definition in a Create Table statement.

- You cannot specify ANYTYPE, BINARY, COMBOBOX, or TIME data types in the column definition of Alter Table statements.

- If a SQL view includes `SELECT * FROM` for the table to which the column was added in the view's Select statement, the new column is added to the view.

Example A
Assuming a schema named SFORCE, this example adds the `status` column with a default value of ACTIVE to the `test` table.

```
ALTER TABLE test ADD COLUMN status TEXT(30) DEFAULT 'ACTIVE'
```

The view selects the name column from both the opportunity and account tables. These columns are assigned the alias OpportunityName and AccountName, respectively.

Example B
Assuming a schema named SFORCE, this example adds a `deptId` column that can be used as a foreign key column.

```
ALTER TABLE test ADD COLUMN status TEXT(30) DEFAULT 'ACTIVE'
```
Add Clause: Constraints

Purpose
Supported only for Salesforce-based data stores. Adds a constraint to an existing table. This clause is optional.

Syntax
ADD [CONSTRAINT constraint_name] ...

Notes
• The only type of constraint you can add is a foreign key constraint.
• When adding a foreign key constraint, the table that contains the foreign key must be empty.

Example
Assuming a schema named SFORCE, a foreign key constraint is added to the deptId column of the test table, referencing the rowId of the dept table. For the operation to succeed, the dept table must be empty.
ALTER TABLE test ADD FOREIGN KEY (deptId) REFERENCES dept(rowId)

Drop Clause: Columns

Purpose
Supported only for Salesforce-based data stores. Use the Drop clause to drop a column from an existing table. This clause is optional.

Syntax
DROP {COLUMN} column_name

where:

    column_name

    Specifies an existing column in an existing table.

Notes
• The column being dropped cannot have a constraint defined on it.
• Drop fails if a SQL view includes the column.

Example
This example drops the status column. For the operation to succeed, the status column cannot have a constraint defined on it and cannot be used in a SQL view.
ALTER TABLE test DROP COLUMN status
Create Table for Salesforce

**Purpose**
Creates a new table in the data store.

**Syntax**
```
CREATE TABLE table_name (column_definition [, ...]
[ , constraint_definition ...])
```

where:

- **table_name** specifies the name of the new table. The table name can be qualified by a schema name using the format `schema.table`. If the schema is not specified, the table is created in the current schema.

- **column_definition** specifies the definition of a column in the new table.

- **constraint_definition** specifies constraints on the columns of the new table.

**Notes**
- Creating a table and its relationships can take several minutes.

Column Definition

**Purpose**
Supported only for Salesforce-based data stores. Defines a table column.

**Syntax**
```
column_name Datatype [(precision[,scale])...] 
[DEFAULT default_value][[NOT]NULL][EXT_ID][PRIMARY KEY] 
[START WITH starting_value]  
```

where:

- **column_name** is the name to be assigned to the column.

- **Datatype**
  - is the data type of the column to be created. See Supported data types on page 910 for a list of supported data types. You cannot specify ANYTYPE, BINARY, COMBOBOX, ENCRYPTEDTEXT, or TIME data types in the column definition of Create Table statements.
precision

is the total number of digits for DECIMAL columns, the number of seconds for DATETIME columns, and the length of HTML, LONGTEXTAREA, and TEXT columns.

scale

is the number of digits to the right of the decimal point for DECIMAL columns.

default_value

is the default value to be assigned to the column. The following default values are allowed in column definitions:

• For character columns, a single-quoted string or NULL.
• For datetime columns, a single-quoted Date, Time, or Timestamp value or NULL. You can also use the following datetime SQL functions: CURRENT_DATE, CURRENT_TIMESTAMP, TODAY, or NOW.
• For boolean columns, the literals FALSE, TRUE, NULL.
• For numeric columns, any valid number or NULL.

starting_value

is the starting value for the Identity column. The default start value is 0.

[NOT] NULL

is used to specify whether NULL values are allowed or not allowed in a column. If NOT NULL is specified, all rows in the table must have a column value. If NULL is specified or if neither NULL or NOT NULL is specified, NULL values are allowed in the column.

EXT_ID

is used to specify that the column is an external ID column.

PRIMARY KEY

can only be specified when the data type of the column is ID. ID columns are always the primary key column for Salesforce.

START WITH

specifies the sequence of numbers generated for the Identity column. It can only be used when the data type of the column definition is AUTONUMBER.

Example A

In the following example, the table name is qualified with the schema name, which will create the Test table in the SFORCE schema. The table is created with the following columns: id, Name, and Status. The Status column contains a default value of ACTIVE.

CREATE TABLE SFORCE.Test (id NUMBER(9, 0), Name TEXT(30), Status TEXT(10) DEFAULT 'ACTIVE')
Example B

In the current schema, the following example creates a Test table and gives the id column a starting value of 1000.

```sql
CREATE TABLE Test (id AUTONUMBER START WITH 1000, Name TEXT(30))
```

Example C

The following example creates a dept table with name and deptId columns in the current schema. The deptId column can be used as an external ID column.

```sql
CREATE TABLE dept (name TEXT(30), deptId NUMBER(9, 0) EXT_ID)
```

Constraint Definition

Purpose
Supported only for Salesforce-based data stores. Defines a constraint.

Syntax

```sql
[CONSTRAINT [constraint_name]
{foreign_key_constraint}]
```

where:

- `constraint_name`
  is ignored. The driver uses the data store relationship naming convention to generate the constraint name.

- `foreign_key_constraint`
  defines a link between related tables. See Foreign Key Clause on page 977 for syntax.

A column defined as a foreign key in one table references a primary key in the related table. Only values that are valid in the primary key are valid in the foreign key. The following example is valid because the foreign key values of the dept id column in the EMP table match those of the id column in the referenced table DEPT.

Table 193: Constraint Definition

<table>
<thead>
<tr>
<th>Referenced Table</th>
<th>Main Table</th>
<th>(Foreign Key)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT</td>
<td>EMP</td>
<td></td>
</tr>
<tr>
<td>id</td>
<td>name</td>
<td>id</td>
</tr>
<tr>
<td>1</td>
<td>Dev</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Finance</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Sales</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>dept id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mark</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Jim</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>Mike</td>
<td>2</td>
</tr>
</tbody>
</table>
The following example, however, is not valid. The value 4 in the `dept_id` column does not match any value in the referenced id column of the `DEPT` table.

### Table 194: Constraint Definition

<table>
<thead>
<tr>
<th>Referenced Table</th>
<th>Main Table</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DEPT</code></td>
<td><code>EMP</code></td>
</tr>
<tr>
<td><code>id</code></td>
<td><code>id</code></td>
</tr>
<tr>
<td><code>name</code></td>
<td><code>name</code></td>
</tr>
<tr>
<td><code>dept_id</code></td>
<td><code>dept_id</code></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><code>Dev</code></td>
<td><code>Mark</code></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td><code>Finance</code></td>
<td><code>Jim</code></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td><code>Sales</code></td>
<td><code>Mike</code></td>
</tr>
</tbody>
</table>

### Foreign Key Clause

#### Purpose

Supported only for Salesforce-based data stores. Specifies a foreign key for a constraint.

#### Syntax

```
FOREIGN KEY (fcolumn_name)
REFERENCES ref_table (pcolumn_name)
```

where:

- **fcolumn_name**
  
  Specifies the foreign key column to which the constraint is applied. The data type of this column must be the same as the data type of the column it references.

- **ref_table**
  
  Specifies the table to which the foreign key refers.

- **pcolumn_name**
  
  Specifies the primary key column in the referenced table. For Salesforce, the primary key column is always the `rowId` column.

#### Example

The following example creates the table `emp` with `name, empId,` and `deptId` columns in the current schema. The table contains a foreign key constraint on the `deptId` column, referencing the `rowId` in the `dept` table created in Column Definition on page 974. For the operation to succeed, the data type of the `deptId` column must be the same as that of the `rowId` column.

```sql
CREATE TABLE emp (name TEXT(30), empId NUMBER(9, 0) EXT_ID, deptId TEXT(18), FOREIGN KEY(deptId) REFERENCES dept(rowId))
```
Delete

Purpose
The Delete statement is used to delete rows from a table.

Syntax
DELETE FROM table_name [WHERE search_condition]

where:
table_name specifies the name of the table from which you want to delete rows.
search_condition is an expression that identifies which rows to delete from the table.

The Where clause determines which rows are to be deleted. Without a Where clause, all rows of the table are deleted, but the table is left intact. See Where Clause on page 986 for information about the syntax of Where clauses. Where clauses can contain subqueries.

Example A
This example shows a Delete statement on the emp table.

DELETE FROM emp WHERE emp_id = 'E10001'

Each Delete statement removes every record that meets the conditions in the Where clause. In this case, every record having the employee ID E10001 is deleted. Because employee IDs are unique in the employee table, at most, one record is deleted.

Example B
This example shows using a subquery in a Delete clause.

DELETE FROM emp WHERE dept_id = (SELECT dept_id FROM dept WHERE dept_name = 'Marketing')

The records of all employees who belong to the department named Marketing are deleted.

Drop Table for Salesforce

Purpose
The Drop Table statement drops (removes) a table, its data, and its indexes.

Syntax
DROP TABLE table_name [IF EXISTS] [RESTRICT | CASCADE]

where:
table_name

Specifies the name of an existing table to drop.
IF EXISTS

   Specifies that an error is not to be returned if the table does not exist.

RESTRICT

   Is in effect by default, meaning that the drop fails if any tables or views reference this table.

CASCADE

   Specifies that the drop extends to linked objects and any tables that reference the specified table are dropped also.

Explain Plan

Purpose

The Explain Plan statement can be used with any query to retrieve a detailed list of the elements in the execution plan. Explain Plan generates a result set with a single column named OPERATION. The individual elements that comprise the plan are returned as rows in the result set.

Syntax

EXPLAIN PLAN FOR {SELECT ... | DELETE ... | INSERT ... | UPDATE ...}

The returned list of elements includes the indexes used for performing the query and can be used to optimize the query.

Insert

Purpose

The Insert statement is used to add new rows to a table. You can specify either of the following options:

• List of values to be inserted as a new row
• Select statement that copies data from another table to be inserted as a set of new rows

Syntax

INSERT INTO table_name [(column_name[,column_name]...)]
VALUES (expression [,expression]...) | select_statement

table_name is the name of the table in which you want to insert rows.

column_name is optional and specifies an existing column. Multiple column names (a column list) must be separated by commas. A column list provides the name and order of the columns, the values of which are specified in the Values clause. If you omit a column_name or a column list, the value expressions must provide values for all columns defined in the table and must be in the same order that the columns are defined for the table. Table columns that do not appear in the column list are populated with the default value, or with NULL if no default value is specified. See Specifying an External ID Column on page 980 for more information.

expression is the list of expressions that provides the values for the columns of the new record. Typically, the expressions are constant values for the columns. Character string values must be enclosed in single quotation marks ('). See Literals on page 993 for more information.
select_statement is a query that returns values for each column_name value specified in the column list. Using a Select statement instead of a list of value expressions lets you select a set of rows from one table and insert it into another table using a single Insert statement. The Select statement is evaluated before any values are inserted. This query cannot be made on the table into which values are inserted. See Select on page 981 for information about Select statements.

Specifying an External ID Column

Use the following syntax to specify an external ID column to look up the value of a foreign key column.

Syntax

    column_name EXT_ID [schema_name.[table_name.]] ext_id_column

where:

EXT_ID

    is used to specify that the column specified by ext_id_column is used to look up the rowid to be inserted into the column specified by column_name.

schema_name

    is the name of the schema of the table that contains the foreign key column being specified as the external ID column.

table_name

    is the name of the table that contains the foreign key column being specified as the external ID column.

ext_id_column

    is the external ID column.

Example A

This example uses a list of expressions to insert records. Each Insert statement adds one record to the database table. In this case, one record is added to the table emp. Values are specified for five columns. The remaining columns in the table are assigned the default value or NULL if no default value is specified.

    INSERT INTO emp (last_name, 
                    first_name, 
                    emp_id, 
                    salary, 
                    hire_date) 
    VALUES ('Smith', 'John', 'E22345', 27500, {1999-04-06})

Example B

This example uses a Select statement to insert records. The number of columns in the result of the Select statement must match exactly the number of columns in the table if no column list is specified, or it must match the number of column names specified in the column list. A new entry is created in the table for every row of the Select result.

    INSERT INTO empl (first_name, 
                      last_name, 
                      ...
emp_id, 
  dept, 
  salary)
SELECT first_name, last_name, emp_id, dept, salary FROM emp
WHERE dept = 'D050'

Example C
This example uses a list of expressions to insert records and specifies an external ID column (a foreign key column) named accountId that references a table that has an external ID column named AccountNum.

INSERT INTO emp (last_name, 
  first_name, 
  emp_id, 
  salary, 
  hire_date, 
  accountId EXT_ID AccountNum)
VALUES ('Smith', 'John', 'E22345', 27500, {1999-04-06}, 0001)

Select

Purpose
The Select statement is used to fetch results from one or more tables.

Syntax
SELECT select_clause 
from_clause 
[where_clause] 
[groupby_clause] 
[having_clause] 
[GROUP BY[ DISTINCT] | 
INTERSECT [DISTINCT]] select_statement] 
[orderby_clause] 
[limit_clause]

where:
select_clause
specifies the columns from which results are to be returned by the query. See Select Clause on page 982 for a complete explanation.

from_clause
specifies one or more tables on which the other clauses in the query operate. See From Clause on page 984 for a complete explanation.

where_clause
is optional and restricts the results that are returned by the query. See Where Clause on page 986 for a complete explanation.
groupby_clause

is optional and allows query results to be aggregated in terms of groups. See Group By Clause on page 986 for a complete explanation.

having_clause

is optional and specifies conditions for groups of rows (for example, display only the departments that have salaries totaling more than $200,000). See Having Clause on page 987 for a complete explanation.

UNION

is an optional operator that combines the results of the left and right Select statements into a single result. See Union Operator on page 988 for a complete explanation.

INTERSECT

is an optional operator that returns a single result by keeping any distinct values from the results of the left and right Select statements. See Intersect Operator on page 989 for a complete explanation.

EXCEPT | MINUS

are synonymous optional operators that returns a single result by taking the results of the left Select statement and removing the results of the right Select statement. See Except and Minus Operators on page 989 for a complete explanation.

orderby_clause

is optional and sorts the results that are returned by the query. See Order By Clause on page 990 for a complete explanation.

limit_clause

is optional and places an upper bound on the number of rows returned in the result. See Limit Clause on page 991 for a complete explanation.

Select Clause

The Select clause is used to determine the columns you want to retrieve by specifying column expressions, or all columns by specifying an asterisk (*).

Syntax

```
SELECT [{LIMIT offset number | TOP number}] [ALL | DISTINCT] {* | column_expression
[[AS] column_alias] [, column_expression [[AS] column_alias], ...])
[INTO [DISK | TEMP] new_table]
SELECT [{LIMIT offset limit | TOP limit}] [ALL | DISTINCT]
{select_expression | table.* | *} [, ...]
[INTO [DISK | TEMP] new_table]
```

where:

LIMIT offset number creates the result set for the Select statement first and then discards the first number of rows specified by offset and returns the number of remaining rows specified by number. To not discard any of the rows, specify 0 for offset, for example, LIMIT 0 number. To discard the first offset number of rows and return all the remaining rows, specify 0 for number, for example, LIMIT offset0.


TOP number is equivalent to LIMIT 0 number.

column expression can be simply a column name (for example, last_name). More complex expressions may include mathematical operations or string manipulation (for example, salary * 1.05). See SQL Expressions on page 992 for details. column_expression can also include aggregate functions. See Aggregate Functions on page 983 for details.

column_alias can be used to give the column a descriptive name. For example, to assign the alias department to the column dep:

```
SELECT dep AS department FROM emp
```

Separate multiple column expressions with commas (for example, SELECT last_name, first_name, hire_date).

Column names can be prefixed with the table name or table alias. For example, SELECT emp.last_name or e.last_name, where e is the alias for the table emp.

The DISTINCT operator can precede the first column expression. This operator eliminates duplicate rows from the result of a query. For example:

```
SELECT DISTINCT dep FROM emp
```

NULL values are not treated as distinct from each other. The default behavior is that all result rows be returned, which can be made explicit with the keyword ALL.

The INTO clause copies the result set into new_table. INTO DISK creates the new table in cached memory. INTO TEMP creates a temporary table.

Notes

- Separate multiple column expressions with commas (for example, SELECT last_name, first_name, hire_date).
- Column names can be prefixed with the table name or table alias. For example, SELECT emp.last_name or e.last_name, where e is the alias for the table emp.
- NULL values are not treated as distinct from each other. The default behavior is that all result rows be returned, which can be made explicit with the keyword ALL.

Aggregate Functions

Aggregate functions can also be a part of a Select clause. Aggregate functions return a single value from a set of rows. An aggregate can be used with a field name (for example, AVG(SALARY)) or in combination with a more complex column expression (for example, AVG(SALARY * 1.07)). The column expression can be preceded by the DISTINCT operator. The DISTINCT operator eliminates duplicate values from an aggregate expression. For example:

```
COUNT (DISTINCT last_name)
```

In this example, only distinct last name values are counted.

The following table lists valid aggregate functions.

<table>
<thead>
<tr>
<th>Aggregate</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM</td>
<td>The total of the values in a numeric field expression. For example, SUM(SALARY) returns the sum of all salary field values.</td>
</tr>
</tbody>
</table>
Returns

<table>
<thead>
<tr>
<th>Aggregate</th>
<th>Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AVG</strong></td>
<td>The average of the values in a numeric field expression. For example, ( \text{AVG(SALARY)} ) returns the average of all salary field values.</td>
</tr>
<tr>
<td><strong>COUNT</strong></td>
<td>The number of values in any field expression. For example, ( \text{COUNT(NAME)} ) returns the number of name values. When using ( \text{COUNT} ) with a field name, ( \text{COUNT} ) returns the number of non-NULL field values. A special example is ( \text{COUNT(*)} ), which returns the number of rows in the set, including rows with NULL values.</td>
</tr>
<tr>
<td><strong>MAX</strong></td>
<td>The maximum value in any field expression. For example, ( \text{MAX(SALARY)} ) returns the maximum salary field value.</td>
</tr>
<tr>
<td><strong>MIN</strong></td>
<td>The minimum value in any field expression. For example, ( \text{MIN(SALARY)} ) returns the minimum salary field value.</td>
</tr>
</tbody>
</table>

**From Clause**

**Purpose**

The **From** clause indicates the tables to be used in the **Select** statement.

**Syntax**

```
FROM table_name [table_alias] [...] 
```

where:

- **table_name**
  
  Is the name of a table or a subquery. Multiple tables define an implicit inner join among those tables. Multiple table names must be separated by a comma. For example:

  ```
  SELECT * FROM emp, dep
  ```

  Subqueries can be used instead of table names. Subqueries must be enclosed in parentheses. See Subquery in a From Clause on page 986 for an example.

- **table_alias**
  
  Is a name used to refer to a table in the rest of the Select statement. When you specify an alias for a table, you can prefix all column names of that table with the table alias.

**Example**

The following example specifies two table aliases, \( e \) for **emp** and \( d \) for **dep**:

```sql
SELECT e.name, d.deptName
FROM emp e, dep d
WHERE e.deptId = d.id
```
**table_alias** is a name used to refer to a table in the rest of the Select statement. When you specify an alias for a table, you can prefix all column names of that table with the table alias. For example, given the table specification:

```sql
FROM emp E
```

you may refer to the last_name field as E.last_name. Table aliases must be used if the Select statement joins a table to itself. For example:

```sql
SELECT * FROM emp E, emp F WHERE E.mgr_id = F.emp_id
```

The equal sign (=) includes only matching rows in the results.

**Outer Join Escape Sequences**

JDBC supports the SQL-92 left, right, and full outer join syntax. The escape sequence for outer joins is:

```sql
{oj outer-join}
```

where **outer-join** is

```sql
table-reference \{LEFT | RIGHT | FULL\} OUTER JOIN \{table-reference | outer-join\} ON search-condition
```

where **table-reference** is a database table name, and **search-condition** is the join condition you want to use for the tables.


The following outer join escape sequences are supported by Salesforce data stores:

- Left outer joins
- Right outer joins
- Nested outer joins

**Join in a From Clause**

You can use a Join as a way to associate multiple tables within a Select statement. Joins may be either explicit or implicit. For example, the following is the example from the previous section restated as an explicit inner join:

```sql
SELECT * FROM emp INNER JOIN dep ON id=empId
SELECT e.name, d.deptName
FROM emp e INNER JOIN dep d ON e.deptId = d.id;
```

whereas the following is the same statement as an implicit inner join:

```sql
SELECT * FROM emp, dep
```

**Syntax**

```sql
FROM table_name \{RIGHT OUTER | INNER | LEFT OUTER | CROSS\} JOIN table.key ON search-condition
```
Example
In this example, two tables are joined using LEFT OUTER JOIN. T1, the first table named includes nonmatching rows.

```sql
SELECT * FROM T1 LEFT OUTER JOIN T2 ON T1.key = T2.key
```

If you use a CROSS JOIN, no ON expression is allowed for the join.

Subquery in a From Clause
Subqueries can be used in the From clause in place of table references (table_name). For example:

```sql
SELECT * FROM (SELECT * FROM emp WHERE sal > 10000) new_emp, dept WHERE new_emp.deptno = dept.deptno
```

Where Clause

Purpose
Specifies the conditions that rows must meet to be retrieved.

Syntax

```
WHERE expr1 rel_operator expr2
```

where:

- `expr1`
  - is either a column name, literal, or expression.

- `expr2`
  - is either a column name, literal, expression, or subquery. Subqueries must be enclosed in parentheses.

- `rel_operator`
  - is the relational operator that links the two expressions.

Example
This Select statement retrieves the first and last names of employees that make at least $20,000.

```sql
SELECT last_name, first_name FROM emp WHERE salary >= 20000
```

See also

- Subqueries on page 1000
- SQL Expressions on page 992

Group By Clause

Purpose
Specifies the names of one or more columns by which the returned values are grouped. This clause is used to return a set of aggregate values.
Syntax

GROUP BY column_expression [, ...]

where:

column_expression

is either a column name or a SQL expression. Multiple values must be separated by a comma. If
column_expression is a column name, it must match one of the column names specified in the
Select clause. Also, the Group By clause must include all non-aggregate columns specified in
the Select list.

Example

The following example totals the salaries in each department:

SELECT dept_id, sum(salary) FROM emp GROUP BY dept_id

This statement returns one row for each distinct department ID. Each row contains the department ID and the
sum of the salaries of the employees in the department.

See also

Subqueries on page 1000
SQL Expressions on page 992

Having Clause

Purpose

Specifies conditions for groups of rows (for example, display only the departments that have salaries totaling
more than $200,000). This clause is valid only if you have already defined a Group By clause.

Syntax

HAVING expr1 rel_operator expr2

where:

expr1

is a column name, a constant value, or an expression. An expression does not have to match a
column expression in the Select clause.

expr2

is a column name, a constant value, or an expression. An expression does not have to match a
column expression in the Select clause.

rel_operator

is the relational operator that links the two expressions.
Example
This example returns only the departments that have salaries totaling more than $200,000:

```
SELECT dept_id, sum(salary) FROM emp
GROUP BY dept_id HAVING sum(salary) > 200000
```

See also
Subqueries on page 1000
SQL Expressions on page 992

Union Operator

Purpose
Combines the results of two `Select` statements into a single result. The single result is all the returned rows from both Select statements. By default, duplicate rows are not returned. To return duplicate rows, use the All keyword (`UNION ALL`).

Syntax
```
select_statement
UNION [ALL | DISTINCT] | {MINUS [DISTINCT] | EXCEPT [DISTINCT]} | INTERSECT
[DISTINCT]
select_statement
```

Notes
- When using the `Union` operator, the `Select` lists for each `Select` statement must have the same number of column expressions with the same data types and must be specified in the same order.

Example A
The following example has the same number of column expressions, and each column expression, in order, has the same data type.

```
SELECT last_name, salary, hire_date FROM emp
UNION
SELECT name, pay, birth_date FROM person
```

Example B
The following example is not valid because the data types of the column expressions are different (`salary FROM emp` has a different data type than `last_name FROM raises`). This example does have the same number of column expressions in each `Select` statement but the expressions are not in the same order by data type.

```
SELECT last_name, salary FROM emp
UNION
SELECT salary, last_name FROM raises
```
Intersect Operator

Purpose
Returns a single result set. The result set contains rows that are returned by both Select statements. Duplicates are returned unless the DISTINCT operator is added.

Syntax
```
select_statement
INTERSECT [DISTINCT]
select_statement
```
DISTINCT eliminates duplicate rows from the results.

Notes
• When using the INTERSECT operator, the Select lists for each Select statement must have the same number of column expressions with the same data types and must be specified in the same order.

Example A
The following example has the same number of column expressions, and each column expression, in order, has the same data type.

```
SELECT last_name, salary, hire_date FROM emp
INTERSECT [DISTINCT]
SELECT name, pay, birth_date FROM person
```

Example B
The following example is not valid because the data types of the column expressions are different (salary FROM emp has a different data type than last_name FROM raises). This example does have the same number of column expressions in each Select statement but the expressions are not in the same order by data type.

```
SELECT last_name, salary FROM emp
UNION
SELECT salary, last_name FROM raises
```

Except and Minus Operators

Purpose
Returns the rows from the left Select statement that are not included in the result of the right Select statement. These operators are synonymous.

Syntax
```
select_statement
{EXCEPT [DISTINCT] | MINUS [DISTINCT]}
select_statement
```
DISTINCT eliminates duplicate rows from the results.
Notes

• When using one of these operators, the Select lists for each Select statement must have the same number of column expressions with the same data types and must be specified in the same order.

Example A

The following example has the same number of column expressions, and each column expression, in order, has the same data type.

```
SELECT last_name, salary, hire_date FROM emp
EXCEPT
SELECT name, pay, birth_date FROM person
```

Example B

The following example is not valid because the data types of the column expressions are different (salary FROM emp has a different data type than last_name FROM raises). This example does have the same number of column expressions in each Select statement but the expressions are not in the same order by data type.

```
SELECT last_name, salary FROM emp
EXCEPT
SELECT salary, last_name FROM raises
```

Order By Clause

Purpose

Specifies how the rows are to be sorted.

Syntax

```
ORDER BY sort_expression [DESC | ASC] [, ...]
```

where:

```
sort_expression
```

is either the name of a column, a column alias, a SQL expression, or the positioned number of the column or expression in the select list to use.

The default is to perform an ascending (ASC) sort.

Example

To sort by last_name and then by first_name, you could use either of the following Select statements:

```
SELECT emp_id, last_name, first_name FROM emp
ORDER BY last_name, first_name
```

or

```
SELECT emp_id, last_name, first_name FROM emp
ORDER BY 2, 3
```

In the second example, last_name is the second item in the Select list, so ORDER BY 2, 3 sorts by last_name and then by first_name.
See also
Subqueries on page 1000
SQL Expressions on page 992

Limit Clause

Purpose
Places an upper bound on the number of rows returned in the result.

Syntax
LIMIT number_of_rows [OFFSET offset_number]

where:

number_of_rows
    specifies a maximum number of rows in the result. A negative number indicates no upper bound.

OFFSET
    specifies how many rows to skip at the beginning of the result set. offset_number is the number of rows to skip.

Notes
• In a compound query, the Limit clause can appear only on the final Select statement. The limit is applied to the entire query, not to the individual Select statement to which it is attached.

Example
The following example returns a maximum of 20 rows.

SELECT last_name, first_name FROM emp WHERE salary > 20000 ORDER BY dept_id
LIMIT 20

Update

Purpose
An Update statement changes the value of columns in selected rows of a table.

Syntax
UPDATE table_name SET column_name = expression
[, column_name = expression] [WHERE conditions]

table_name
    Is the name of the table for which you want to update values.
column_name

Is the name of a column, the value of which is to be changed. Multiple column values can be changed in a single statement.

expression

Is the new value for the column. The expression can be a constant value or a subquery that returns a single value. Subqueries must be enclosed in parentheses.

Notes

• A Where clause can be used to restrict which rows are updated.

See also

Subqueries on page 1000
Where Clause on page 986

Example A

The following example changes every record that meets the conditions in the Where clause. In this case, the salary and exempt status are changed for all employees having the employee ID E10001. Because employee IDs are unique in the emp table, only one record is updated.

UPDATE emp SET salary=32000, exempt=1 WHERE emp_id = 'E10001'

Example B

The following example uses a subquery. In this example, the salary is changed to the average salary in the company for the employee having employee ID E10001.

UPDATE emp SET salary = (SELECT avg(salary) FROM emp) WHERE emp_id = 'E10001'

SQL Expressions

Each data store supports a number of SQL expressions. An expression is a combination of one or more values, operators, and SQL functions that evaluate to a value. You can use expressions in the Where, Having, and Order By clauses of Select statements; and in the Set clauses of Update statements.

Expressions enable you to use mathematical operations as well as character string manipulation operators to form complex queries.

Hybrid Data Pipeline supports both unquoted and quoted identifiers. An unquoted identifier must start with an ASCII alpha character and can be followed by zero or more ASCII alphanumeric characters. Unquoted identifiers are converted to uppercase before being used.

Quoted identifiers must be enclosed in double quotation marks (" "). A quoted identifier can contain any Unicode character including the space character. The Hybrid Data Pipeline service recognizes the Unicode escape sequence \uxxxx as a Unicode character. You can specify a double quotation mark in a quoted identifier by escaping it with a double quotation mark.

The maximum length of both quoted and unquoted identifiers is 128 characters.

Valid expression elements are:

• Column names: The most common expression is a simple column name. You can combine a column name with other expression elements.
• **Literals**: Literals are fixed data values. See [Literals](#) on page 993 for more information.

• **Operators**: An operator manipulates individual data items and returns a result. See [Operators](#) on page 995 for more information.

• **Functions**: Hybrid Data Pipeline supports a number of functions that you may use in expressions, as listed and described in the [Supported scalar functions](#) on page 940 section.

• **Conditions**: A condition specifies a combination of one or more expressions and logical operators that evaluates to either TRUE, FALSE, or UNKNOWN. See [Conditions](#) on page 999 for more information.

**Literals**

Literals are fixed data values. For example, in the expression `PRICE * 1.05`, the value `1.05` is a constant. Literals are classified into types, including the following:

- Binary
- Character string
- Date
- Floating point
- Integer
- Numeric
- Time
- Timestamp

The following table describes the literal format for supported SQL data types.

**Table 196: Literal Syntax Examples**

<table>
<thead>
<tr>
<th>SQL Type</th>
<th>Literal Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIGINT</td>
<td>( n ) where ( n ) is any valid integer value in the range of the <code>INTEGER</code> data type</td>
<td>12 or -34 or 0</td>
</tr>
</tbody>
</table>
| BOOLEAN      | Min Value: 0  
Max Value: 1                                    | 0 or 1                  |
| DATE         | 'yyyy-mm-dd'                                                                | '2010-05-21'            |
| DATETIME     | 'yyyy-mm-dd hh:mm:ss.SSSSSS'                                                | '2010-05-21 18:33:05.025' |
| DECIMAL      | \( n.f \) where \( n \) is the integral part and \( f \) is the fractional part | 0.25 or 3.1415 or -7.48 |
| DOUBLE       | \( n.fE\times \) where \( n \) is the integral part, \( f \) is the fractional part, and \( \times \) is the exponent | 1.2E0 or 2.5E40 or -3.45E2 or 5.67E-4 |
| INTEGER      | \( n \) where \( n \) is a valid integer value in the range of the `INTEGER` data type | 12 or -34 or 0           |
### Character String Literals

Text specifies a character string literal. A character string literal must be enclosed in single quotation marks. To represent one single quotation mark within a literal, you must enter two single quotation marks. When the data in the fields is returned to the client, trailing blanks are stripped.

A character string literal can have a maximum length of 32 KB, that is, (32*1024) bytes.

**Example**

'Hello'
'Jim''s friend is Joe'

### Integer Literals

Integer literals are represented by a string of numbers that are not enclosed in quotation marks and do not contain decimal points.

**Note:**

- Integer constants must be whole numbers; they cannot contain decimals.
- Integer literals can start with sign characters (+/−).

**Example**

1994 or -2

### Numeric Literals

Unquoted numeric values are treated as numeric literals. If the unquoted numeric value contains a decimal point or exponent, it is treated as a real literal; otherwise, it is treated as an integer literal.

**Example**

+1894.1204
Binary Literals
Binary literals are represented with single quotation marks. The valid characters in a binary literal are 0-9, a-f, and A-F.

Example
'00af123d'

Date/Time Literals
Date and time literal values are:
- A Date literal is enclosed in single quotation marks (' '). The format is yyyy-mm-dd.
- A Time literal is enclosed in single quotation marks (' '). The format is hh:mm:ss.
- A Timestamp is enclosed in single quotation marks (' '). The format is yyyy-mm-dd hh:mm:ss.SSSSSS.

Operators
This section describes the operators that can be used in SQL expressions.

Unary Operator
A unary operator operates on only one operand.

Syntax
operator operand

Binary Operator
A binary operator operates on two operands.

Syntax
operand1 operator operand2

If an operator is given a null operand, the result is always null. The only operator that does not follow this rule is concatenation (||), which always returns a VARCHAR.

Arithmetic Operator
You can use an arithmetic operator in an expression to negate, add, subtract, multiply, and divide numeric values. The result of this operation is also a numeric value. The + and - operators are also supported in date/time fields to allow date arithmetic. The following table lists the supported arithmetic operators.
Table 197: Arithmetic Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ -</td>
<td>Denotes a positive or negative expression. These are unary operators.</td>
<td>SELECT * FROM emp WHERE comm = -1</td>
</tr>
<tr>
<td>* /</td>
<td>Multiplies, divides. These are binary operators.</td>
<td>UPDATE emp SET sal = sal + sal * 0.10</td>
</tr>
<tr>
<td>+ -</td>
<td>Adds, subtracts. These are binary operators.</td>
<td>SELECT sal + comm FROM emp WHERE empno &gt; 100</td>
</tr>
</tbody>
</table>

Concatenation Operator

The concatenation operator manipulates character strings. The following table lists the only supported concatenation operator.

Table 198: Concatenation Operator

<table>
<thead>
<tr>
<th>Operator</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The result of concatenating two character strings is the data type VARCHAR.

Comparison Operator

Comparison operators compare one expression to another. The result of such a comparison can be TRUE, FALSE, or UNKNOWN (if one of the operands is NULL). The Hybrid Data Pipeline driver considers the UNKNOWN result as FALSE.

The following table lists the supported comparison operators.
<table>
<thead>
<tr>
<th>Operator</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equality test.</td>
<td>SELECT * FROM emp WHERE sal = 1500</td>
</tr>
<tr>
<td>!= or &lt;&gt;</td>
<td>Inequality test.</td>
<td>SELECT * FROM emp WHERE sal != 1500 or sal &lt;&gt; 1500</td>
</tr>
</tbody>
</table>
| > and <          | "Greater than" and "less than" tests. | SELECT * FROM emp WHERE sal > 1500  
SELECT * FROM emp WHERE sal < 1500 |
| >= and <=        | "Greater than or equal to" and "less than or equal to" tests. | SELECT * FROM emp WHERE sal >= 1500  
SELECT * FROM emp WHERE sal <= 1500 |
| [NOT] IN         | "Equal to any member of" test. | SELECT * FROM emp WHERE job IN ('CLERK', 'ANALYST')  
SELECT * FROM emp WHERE sal IN (SELECT sal FROM emp WHERE deptno = 30) |
| [NOT] BETWEEN x AND y | "Greater than or equal to x" and "less than or equal to y." | SELECT * FROM emp WHERE sal BETWEEN 2000 AND 3000 |
| EXISTS           | Tests for existence of rows in a subquery. | SELECT empno, ename, deptno FROM emp e WHERE EXISTS  
(SELECT deptno FROM dept WHERE e.deptno = dept.deptno) |
| IS [NOT] NULL    | Tests whether the value of the column or expression is NULL. | SELECT * FROM emp WHERE ename IS NOT NULL  
SELECT * FROM emp WHERE ename IS NULL |
| ESCAPE clause in LIKE operator LIKE 'pattern string' ESCAPE 'c' | | |
Logical Operator

A logical operator combines the results of two component conditions to produce a single result or to invert the result of a single condition. The following table lists the supported logical operators.

Table 200: Logical Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOT</td>
<td>Returns TRUE if the following condition is FALSE. Returns FALSE if it is TRUE. If it is UNKNOWN, it remains UNKNOWN.</td>
<td></td>
</tr>
<tr>
<td>AND</td>
<td>Returns TRUE if both component conditions are TRUE. Returns FALSE if either is FALSE; otherwise, returns UNKNOWN.</td>
<td></td>
</tr>
<tr>
<td>OR</td>
<td>Returns TRUE if either component condition is TRUE. Returns FALSE if both are FALSE; otherwise, returns UNKNOWN.</td>
<td></td>
</tr>
</tbody>
</table>

Example

In the Where clause of the following Select statement, the AND logical operator is used to ensure that managers earning more than $1000 a month are returned in the result:

```sql
SELECT * FROM emp WHERE jobtitle = manager AND sal > 1000
```

Operator Precedence

As expressions become more complex, the order in which the expressions are evaluated becomes important. The following table shows the order in which the operators are evaluated. The operators in the first line are evaluated first, then those in the second line, and so on. Operators in the same line are evaluated left to right in the expression. You can change the order of precedence by using parentheses. Enclosing expressions in parentheses forces them to be evaluated together.
Table 201: Operator Precedence

<table>
<thead>
<tr>
<th>Precedence</th>
<th>Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+ (Positive), - (Negative)</td>
</tr>
<tr>
<td>2</td>
<td>* (Multiply), / (Division)</td>
</tr>
<tr>
<td>3</td>
<td>+ (Add), - (Subtract)</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>=, &gt;, &lt;, &gt;=, &lt;=, &lt;&gt;, != (Comparison operators)</td>
</tr>
<tr>
<td>6</td>
<td>NOT, IN, LIKE</td>
</tr>
<tr>
<td>7</td>
<td>AND</td>
</tr>
<tr>
<td>8</td>
<td>OR</td>
</tr>
</tbody>
</table>

Example A

The query in the following example returns employee records for which the department number is 1 or 2 and the salary is greater than $1000:

```sql
SELECT * FROM emp WHERE (deptno = 1 OR deptno = 2) AND sal > 1000
```

Because parenthetical expressions are forced to be evaluated first, the OR operation takes precedence over AND.

Example B

In the following example, the query returns records for all the employees in department 1, but only employees whose salary is greater than $1000 in department 2.

```sql
SELECT * FROM emp WHERE deptno = 1 OR deptno = 2 AND sal > 1000
```

The AND operator takes precedence over OR, so that the search condition in the example is equivalent to the expression `deptno = 1 OR (deptno = 2 AND sal > 1000)`.

**Conditions**

A condition specifies a combination of one or more expressions and logical operators that evaluates to either TRUE, FALSE, or UNKNOWN. You can use a condition in the Where clause of the Delete, Select, and Update statements; and in the Having clauses of Select statements. The following describes supported conditions.

Table 202: Conditions

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple comparison</td>
<td>Specifies a comparison with expressions or subquery results.</td>
</tr>
<tr>
<td></td>
<td>=, !=, &lt;&gt;, &lt;, &gt;, &lt;=, &lt;=</td>
</tr>
</tbody>
</table>
### Condition | Description
--- | ---
Group comparison | Specifies a comparison with any or all members in a list or subquery. 
| | \([- =, !=, <>, <, >, <=, <=] \{ANY, ALL, SOME\}\)
Membership | Tests for membership in a list or subquery. 
| | \([NOT] \text{IN}\)
Range | Tests for inclusion in a range. 
| | \([NOT] \text{BETWEEN}\)
NULL | Tests for nulls. 
| | \(\text{IS NULL, IS NOT NULL}\)
EXISTS | Tests for existence of rows in a subquery. 
| | \([NOT] \text{EXISTS}\)
LIKE | Specifies a test involving pattern matching. 
| | \([NOT] \text{LIKE}\)
Compound | Specifies a combination of other conditions. 
| | \(\text{CONDITION [AND/OR] CONDITION}\)

### Subqueries

A query is an operation that retrieves data from one or more tables or views. In this reference, a top-level query is called a Select statement, and a query nested within a Select statement is called a subquery.

A subquery is a query expression that appears in the body of another expression such as a Select, an Update, or a Delete statement. In the following example, the second Select statement is a subquery:

```sql
SELECT * FROM emp WHERE deptno IN
(SELECT deptno FROM dept)
```

### IN Predicate

The In predicate specifies a set of values against which to compare a result set. If the values are being compared against a subquery, only a single column result set is returned.
Syntax

```
value [NOT] IN (value1, value2,...)  
OR value [NOT] IN (subquery)
```

Example

```
SELECT * FROM emp WHERE deptno IN  
(SELECT deptno FROM dept WHERE dname <> 'Sales')
```

**EXISTS Predicate**

The `EXISTS` predicate is true only if the cardinality of the subquery is greater than 0; otherwise, it is false.

Syntax

```
EXISTS (subquery)
```

Example

```
SELECT empno, ename, deptno FROM emp e WHERE EXISTS  
(SELECT deptno FROM dept WHERE e.deptno = dept.deptno)
```

**UNIQUE Predicate**

The `UNIQUE` predicate is used to determine whether duplicate rows exist in a virtual table (one returned from a subquery).

Syntax

```
UNIQUE (subquery)
```

Example

```
SELECT * FROM dept d WHERE UNIQUE  
(SELECT deptno FROM emp e WHERE e.deptno = d.deptno)
```

**Correlated Subqueries**

**Purpose**

A correlated subquery is a subquery that references a column from a table referred to in the parent statement. A correlated subquery is evaluated once for each row processed by the parent statement. The parent statement can be a `Select`, `Update`, or `Delete` statement.

A correlated subquery answers a multiple-part question in which the answer depends on the value in each row processed by the parent statement. For example, you can use a correlated subquery to determine which employees earn more than the average salaries for their departments. In this case, the correlated subquery specifically computes the average salary for each department.

**Syntax**

```
SELECT select_list  
FROM table1 t_alias1  
WHERE expr rel_operator  
(SELECT column_list
```
FROM table2 t_alias2
WHERE t_alias1.column =
(SELECT expr
FROM table2 t_alias2
WHERE t_alias1.column = t_alias2.column)
DELETE FROM table1 t_alias1
WHERE column =
(SELECT expr
FROM table2 t_alias2
WHERE t_alias1.column = t_alias2.column)

Notes

• Correlated column names in correlated subqueries must be explicitly qualified with the table name of the parent.

Example A
The following statement returns data about employees whose salaries exceed their department average. This statement assigns an alias to emp, the table containing the salary information, and then uses the alias in a correlated subquery:

```
SELECT deptno, ename, sal FROM emp x WHERE sal >
(SELECT AVG(sal) FROM emp WHERE x.deptno = deptno)
ORDER BY deptno
```

Example B
This is an example of a correlated subquery that returns row values:

```
SELECT * FROM dept "outer" WHERE 'manager' IN
(SELECT managername FROM emp
WHERE "outer".deptno = emp.deptno)
```

Example C
This is an example of finding the department number (deptno) with multiple employees:

```
SELECT * FROM dept main WHERE 1 <
(SELECT COUNT(*) FROM emp WHERE deptno = main.deptno)
```

Example D
This is an example of correlating a table with itself:

```
SELECT deptno, ename, sal FROM emp x WHERE sal >
(SELECT AVG(sal) FROM emp WHERE x.deptno = deptno)
```

Catalog tables

Hybrid Data Pipeline provides a standard set of catalog tables that maintain the information returned by various catalog functions such as SQLTables, SQLColumns, SQLDescribeParam, and SQLDescribeCol. If possible, use the catalog functions to obtain this information instead of querying the catalog tables directly. The INFORMATION_SCHEMA contains additional catalog tables that maintain metadata.
Hybrid Data Pipeline provides catalog tables for the following data store types.

- Salesforce.com
- Force.com
- FinancialForce
- Google Analytics
- ServiceMax
- Veeva CRM
- Microsoft Dynamics CRM Online
- SugarCRM
- Oracle Marketing Cloud
- Oracle Sales Cloud
- Oracle Service Cloud
- Progress Rollbase

Note: Data stores such as Progress OpenEdge, Oracle, and Microsoft SQL Server do not use Hybrid Data Pipeline catalog tables.

**SYSTEM_SESSIONS catalog table**

The system table named `SYSTEM_SESSIONS` stores information about current system sessions. The values in the `SYSTEM_SESSIONS` table are read-only.

The following table defines the columns of the `SYSTEM_SESSIONS` table.

<table>
<thead>
<tr>
<th>Column</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESSION_ID</td>
<td>INTEGER, NOT NULL</td>
<td>A unique ID that identifies this session. The system function CURSESSIONID( ) returns the session ID associated with the connection.</td>
</tr>
<tr>
<td>CONNECTED</td>
<td>DATETIME, NOT NULL</td>
<td>The date and time the session was established.</td>
</tr>
<tr>
<td>USER_NAME</td>
<td>VARCHAR (128), NOT NULL</td>
<td>The name of the embedded database that the session is using.</td>
</tr>
<tr>
<td>IS_ADMIN</td>
<td>BOOLEAN</td>
<td>For internal use only.</td>
</tr>
<tr>
<td>AUTOCOMMIT</td>
<td>BOOLEAN, NOT NULL</td>
<td>For future use.</td>
</tr>
</tbody>
</table>
**SYSTEM_REMOTE_SESSIONS** catalog table

The system table named **SYSTEM_REMOTE_SESSIONS** stores information about the each of the remote sessions that are active for a given data store. The values in the **SYSTEM_REMOTE_SESSIONS** table are read-only.

The following table defines the columns of the **SYSTEM_REMOTE_SESSIONS** table, which is sorted on the following columns: **SESSION_ID** and **SCHEMA**.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESSION_ID</td>
<td>INTEGER, NOT NULL</td>
<td>The connection (session) id with which the remote session is associated.</td>
</tr>
<tr>
<td>SCHEMA</td>
<td>VARCHAR(128), NOT NULL</td>
<td>The schema name that is mapped to the remote session.</td>
</tr>
<tr>
<td>TYPE</td>
<td>VARCHAR(30), NOT NULL</td>
<td>The remote session type. The current valid type is Salesforce.</td>
</tr>
<tr>
<td>Column name</td>
<td>Data type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>INSTANCE</td>
<td>VARCHAR(128)</td>
<td>The remote session instance name or null if the remote data source does not have multiple instances. The Salesforce value for INSTANCE has the following form: Organization_Name [Sandbox] where Organization_Name is the organization name of the instance to which the connection is established. If the connection is established to a sandbox of the organization, then the word Sandbox is added to the end of the name.</td>
</tr>
<tr>
<td>VERSION</td>
<td>VARCHAR(30), NOT NULL</td>
<td>The version of the remote data source to which the session is connected. For Salesforce, this is the version of the Web Service API the driver is using to connect to Salesforce.</td>
</tr>
<tr>
<td>CONFIG_OPTIONS</td>
<td>LONGVARCHAR, NOT NULL</td>
<td>The configuration options used to define the remote data model to relational data model mapping.</td>
</tr>
<tr>
<td>SESSION_OPTIONS</td>
<td>LONGVARCHAR, NOT NULL</td>
<td>The options used to establish the remote connection. This typically is information needed to log into the remote data source. The password value is not displayed.</td>
</tr>
<tr>
<td>WS_CALL_COUNT</td>
<td>INTEGER, NOT NULL</td>
<td>The number of Web service calls made through this remote session. The value of the WS_CALL_COUNT column can be reset using the ALTER SESSION statement.</td>
</tr>
<tr>
<td>WS_AGGREGATE_CALL_COUNT</td>
<td>INTEGER, NOT NULL</td>
<td>The total of all of the Web service calls made to the same remote data source by all active connections using the same server name and user ID.</td>
</tr>
<tr>
<td>REST_AGGREGATE_CALL_COUNT</td>
<td>INTEGER, NOT NULL</td>
<td>The number of REST calls made by this connection. REST calls are used for bulk operations, invoking reports, and describing report parameters.</td>
</tr>
</tbody>
</table>
Error messages

Applications accessing data may encounter error messages, which differ, depending on the data store you are accessing. Each error message is followed by a possible cause and recommended actions, if applicable.

Management API error messages

The following sections describe error messages you may receive back from the Hybrid Data Pipeline Management API. Each error message is followed by a possible cause and recommended actions, if applicable.

In addition to general error messages that apply to all components of the Hybrid Data Pipeline Management, additional error messages are returned only by the Data Source or Connector APIs.

Table 205: Error codes for the Hybrid Data Pipeline Management API

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222206900</td>
<td>Invalid URL for GET: Resource {0} not found.</td>
</tr>
<tr>
<td>222206901</td>
<td>Invalid URL for DELETE: Resource {0} not found.</td>
</tr>
<tr>
<td>222206902</td>
<td>Invalid URL for PUT: Resource {0} not found.</td>
</tr>
<tr>
<td>222206903</td>
<td>Invalid URL for POST: Resource {0} not found.</td>
</tr>
<tr>
<td>222206904</td>
<td>Invalid URL for GET: Resource not specified.</td>
</tr>
<tr>
<td>222206905</td>
<td>Invalid URL for DELETE: Resource not specified.</td>
</tr>
<tr>
<td>222206906</td>
<td>Invalid URL for POST: Resource not specified.</td>
</tr>
<tr>
<td>222206907</td>
<td>Invalid URL for PUT: Resource not specified.</td>
</tr>
<tr>
<td>222206908</td>
<td>The method, {0}, is not allowed for this URL, {1}.</td>
</tr>
<tr>
<td>222206909</td>
<td>Queries are not supported on this call.</td>
</tr>
</tbody>
</table>

HTTP error messages

Hybrid Data Pipeline returns standard HTTP response codes as described in the following table, under the conditions listed in the description.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
</tr>
</tbody>
</table>
|            | The request was successfully completed. If this request created a new resource that is addressable with a URI, and a response body is returned containing a representation of the new resource, a 200 status will be returned with a location header containing the canonical URI for the newly created resource.
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
</tr>
<tr>
<td></td>
<td>A request that created a new resource was completed and no response body containing a representation of the new resource is being returned. A location header containing the canonical URI for the newly created resource will be returned.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
</tr>
<tr>
<td></td>
<td>The JSON request is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Not Authorized</td>
</tr>
<tr>
<td></td>
<td>The user is not authorized. An invalid user name and/or password was used.</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
</tr>
<tr>
<td></td>
<td>This is a client issue, where an application made an illegal request. The server understood the request and is refusing to respond to it.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
</tr>
<tr>
<td></td>
<td>The &lt;DataSource&gt; was not found, where &lt;resource_type&gt; is DataSource.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td></td>
<td>The server encountered an unexpected condition which prevented it from fulfilling the request.</td>
</tr>
<tr>
<td>501</td>
<td>Not Implemented</td>
</tr>
<tr>
<td></td>
<td>The server currently does not support the functionality required to fulfill the request.</td>
</tr>
</tbody>
</table>

**Servlet error messages**

The following section describes error messages you may receive back from an Management API Servlet. Each error message is followed by a possible cause and recommended actions, if applicable.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222206900</td>
<td>Invalid URL for GET: Resource {0} not found.</td>
</tr>
<tr>
<td>222206901</td>
<td>Invalid URL for DELETE: Resource {0} not found.</td>
</tr>
<tr>
<td>222206902</td>
<td>Invalid URL for PUT: Resource {0} not found.</td>
</tr>
<tr>
<td>222206903</td>
<td>Invalid URL for POST: Resource {0} not found.</td>
</tr>
<tr>
<td>222206904</td>
<td>Invalid URL for GET: Resource not specified.</td>
</tr>
<tr>
<td>222206905</td>
<td>Invalid URL for DELETE: Resource not specified.</td>
</tr>
<tr>
<td>222206906</td>
<td>Invalid URL for POST: Resource not specified.</td>
</tr>
</tbody>
</table>
Connector API error messages

The following section describes error messages you may receive back from the Hybrid Data Pipeline Connector API. Each error message is followed by a possible cause and recommended actions, if applicable.

Table 206: Error messages for the Connector API

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222206850</td>
<td>The label {0} is already used by other connector. Please use another label.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The specified label has already been defined by another Connector. The label must be unique.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Modify the label so that it is unique.</td>
</tr>
<tr>
<td>222207100</td>
<td>Problem getting the users from the Access Control List at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207101</td>
<td>Problem adding the user(s) to the Access Control List at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207102</td>
<td>Invalid user name: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The user name in the request payload is not valid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Make sure the user name in the request payload has the appropriate permissions and is specified correctly.</td>
</tr>
<tr>
<td>222207103</td>
<td>There is a problem with the JSON input: Owners -- {0}, {1}--do not match</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The JSON statement is not correct.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the Owners in the JSON input.</td>
</tr>
<tr>
<td>222207104</td>
<td>Problem getting the Connector from the Access Control List at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207106</td>
<td>The number of users specified ((0)) exceeds the system limit ((1)). Please use multiple requests.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: Only one user can be specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Create a separate request for each user.</td>
</tr>
<tr>
<td>222207107</td>
<td>Invalid JSON input: {0}</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The specified JSON input was not valid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Correct the JSON input.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 222207108 | 'authUser' was not supplied or was not an array.  
**Cause:** The request must specify an authUser parameter.  
**Action:** Add an authUser array. The array can be empty. |
| 222207109 | Problem getting the connector info for {0}. Please try again at another time.  
**Cause:** A problem occurred when getting the Connector information for the specified Connector.  
**Action:** Please try again at another time. |
| 222207110 | Problem updating users for {0}. Please try again at another time.  
**Cause:** A problem occurred when updating users for the specified Connector.  
**Action:** Please try again at another time. |
| 222207111 | Problem deleting the user(s) from the Access Control List at this time. Please try again at another time.  
**Cause:** A problem occurred when deleting users from the specified Connector.  
**Action:** Please try again at another time. |
| 222207112 | Connector {0} does not exist or you are not the owner.  
**Cause:** Either the specified On-Premises Connector does not exist, or you are not the owner of the Connector.  
**Action:** The owner specified in the request must match the current owner of the Connector or Connector Group. Changing the owner of a Connector or Connector Group is not supported. |
| 222207115 | Problem getting the Connector info. Please try again at another time.  
**Cause:** A problem occurred when getting the Connector information.  
**Action:** Try the operation later. |
| 222207116 | Problem deleting the Connector.  
**Cause:** A problem occurred deleting the Connector.  
**Action:** Try the operation later. |
| 222207117 | 'members' was not supplied.  
**Cause:** The Connector is a GroupConnector, and must contain a connectorGroup object that contains a members array.  
**Action:** A Connector Group must contain a connectorGroup object that contains a members array. The members array was not defined in the connectorGroup. |
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207118</td>
<td>'memberID' was not supplied.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The <code>members</code> array for this GroupConnector must contain a <code>member_id</code> parameter.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the <code>connectorGroup</code> object. The <code>members</code> array must contain a <code>memberID</code>.</td>
</tr>
<tr>
<td>222207119</td>
<td>'sequence' was not supplied.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The <code>members</code> array for this GroupConnector must contain a <code>sequence</code> parameter.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the <code>connectorGroup</code> object. The <code>members</code> array must contain a <code>sequence</code>.</td>
</tr>
<tr>
<td>222207121</td>
<td>You cannot delete the last member of the Connector Group(s): {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The JSON statement attempted to remove the last member of a Connector Group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> You cannot delete the last member of the Connector Group. To delete a Connector Group, use the Delete Group API.</td>
</tr>
<tr>
<td>222207122</td>
<td>Problem deleting members. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when deleting members from a group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try the operation later.</td>
</tr>
<tr>
<td>222207123</td>
<td>Problem getting members. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when getting members.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try the operation later.</td>
</tr>
<tr>
<td>222207124</td>
<td>ConnectionTimeout must have a value with a minimum of 1.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> ConnectionTimeout wasn't set to a positive integer.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Set ConnectionTimeout to a positive integer, 1 or greater.</td>
</tr>
<tr>
<td>222207125</td>
<td>RetryDelay must have a value with a minimum of 0.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> RetryDelay was set to an invalid value.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Set RetryDelay to 0 or a positive integer. See &quot;Update Connector Information&quot; for more information.</td>
</tr>
<tr>
<td>222207126</td>
<td>There must be at least one member in a Group Connector at all times.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> You attempted to delete the last member of a Group Connector.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> A Group Connector must contain at least one member.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207127</td>
<td>Problem creating a ConnectorId. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: A problem occurred when creating a Connector ID.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207128</td>
<td>This is not a valid payload for an update. Please consult the documentation.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The JSON statement was not valid for an update.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the JSON statement.</td>
</tr>
<tr>
<td>222207129</td>
<td>You cannot change the ConnectorId.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: You cannot change the Connector ID.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: The Connector ID is generated by Hybrid Data Pipeline and is specific to each Connector. It cannot be changed.</td>
</tr>
<tr>
<td>222207130</td>
<td>You cannot change the owner of the Connector.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: You cannot change the owner of the Connector. Only the owner of the Connector can reassign the Connector to a different owner.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Consult the Hybrid Data Pipeline administrator.</td>
</tr>
<tr>
<td>222207131</td>
<td>You cannot add a Group Connector, {0} to another Group.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The specified Connector has been defined as a Group Connector. You cannot add a Group Connector to another Group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Use the Connector ID for a Connector that is not a Group Connector.</td>
</tr>
<tr>
<td>222207132</td>
<td>This Connector {0} is not a member of Connector {1}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The request specified a Connector that is not a member of the Group Connector.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Change the request to use a member of the Group Connector.</td>
</tr>
<tr>
<td>222207133</td>
<td>Problem adding connector(s) to the group connector at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: A problem occurred when adding one or more Connectors to the group connector.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207134</td>
<td>Problem updating connector(s) to the group connector at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: A problem occurred when creating a Connector ID.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| 222207135      | Problem determining authorization to use connector. Please try again at another time.  
**Cause:** A problem occurred when determining authorization to use the Connector.  
**Action:** Try the operation later.                                                                                                                                                                                                                                                                                                                                    |
| 222207136      | Problem getting connector statistics at this time. Please try again at another time.  
**Cause:** A problem occurred when getting connector statistics.  
**Action:** Try the operation later.                                                                                                                                                                                                                                                                                                                                    |
| 222207137      | {0} is not a supported Load Balancing type. Please refer to the documentation on Load Balancing.  
**Cause:** The JSON input specified an invalid type.  
**Action:** See "Enable Round-Robin load balancing for a group" for valid types for load balancing.                                                                                                                                                                                                                                                                                            |
| 222207138      | ConnectorId {0} is already in the following GroupConnector: {1}. A ConnectorId can only be in one GroupConnector.  
**Cause:** The specified Connector is already a member of a Connector group.  
**Action:** Add a different Connector to the Connector group.                                                                                                                                                                                                                                                                                                           |
| 222207139      | Problem updating a ConnectorId. Please try again at another time.  
**Cause:** A problem occurred when updating the Connector information.  
**Action:** Try the operation later.                                                                                                                                                                                                                                                                                                                                    |
| 222207140      | ConnectorId {0} is already in the current GroupConnector {1}. If POST - this GroupConnector failed to be created.  
**Cause:** The specified Connector is already a member of the current Connector group. If you submitted a POST request, the operation failed. The GroupConnector was not created.  
**Action:** Add a different On-Premises Connector to the Connector group.                                                                                                                                                                                                                                                                                        |
| 222207141      | Problem getting version and owner. Please try again at another time.  
**Cause:** A problem occurred when getting version and owner.  
**Action:** Try the operation later.                                                                                                                                                                                                                                                                                                                                    |
| 222207142      | Problem getting authorized users. Please try again at another time.  
**Cause:** A problem occurred when getting authorized users.  
**Action:** Try the operation later.                                                                                                                                                                                                                                                                                                                                    |
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 222207143  | Sequence must be an INTEGER greater than 0.  
**Cause:** The `sequence` parameter must be greater an integer greater than 0.  
**Action:** Check the value of the `sequence` parameter. |
| 222207144  | Weight must be an INTEGER greater than 0.  
**Cause:** The `weight` parameter must be greater an integer greater than 0.  
**Action:** Check the value of the `weight` parameter. |
| 222207145  | Problem getting modified connectors. |
| 222207146  | Connector {0} (Version {1}) does not support Load Balancing. Only version 3.0 and higher support Load Balancing. Please update to the latest version.  
**Cause:** The specified Connector is Version 1.0, and doesn't support Load Balancing.  
**Action:** Update the Connector to Version 3.0 or higher. See the Progress DataDirect Hybrid Data Pipeline Installation Guide for more information. |
| 222207147  | Connector {0} is not a Group. You cannot add Members to a non-Group Connector.  
**Cause:** The request tried to add members to a Connector that is not a Group Connector.  
**Action:** Check the Connector ID. Try the request again using the Connector ID of a Group Connector. |
| 222207148  | Connector {0} is not a Group. You cannot get Members from a non-Group Connector.  
**Cause:** The request tried to get members from a Connector that is not a Group Connector.  
**Action:** Check the Connector ID. Try the request again using the Connector ID of a Group Connector. |
| 222207149  | Connector {0} is not a Group. You cannot delete Members from a non-Group Connector.  
**Cause:** The request tried to delete members from a Connector that is not a Group Connector.  
**Action:** Check the Connector ID. Try the request again using the Connector ID of a Group Connector. |
| 222207150  | You cannot have multiple members with the same sequence.  
**Cause:** The value of the sequence parameter must be unique for each member object.  
**Action:** Change the sequence parameter for one or more members so that each member of the group has a unique value. |
OAuth API error messages

This section describes error messages you may receive from the OAuth API. Each error message is followed by a possible cause and recommended actions, if applicable.

Table 207: Error Messages for the OAuthAPI

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207700</td>
<td>Problem creating an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207701</td>
<td>Problem deleting an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207702</td>
<td>Problem getting OAuthProfiles at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207703</td>
<td>Problem getting an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207704</td>
<td>Problem updating an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207705</td>
<td>Problem creating an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207706</td>
<td>Problem deleting an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The OAuthApplication couldn't be deleted at this time.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try again later.</td>
</tr>
<tr>
<td>222207707</td>
<td>Problem getting OAuthApplications at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207708</td>
<td>Problem getting an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207709</td>
<td>Problem updating an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207710</td>
<td>Invalid OAuthProfileId: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The OAuthProfileId parameter is missing, or no value was defined.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the payload. Add the OAuthProfileId parameter with a valid value.</td>
</tr>
<tr>
<td>222207711</td>
<td>Invalid OAuthApplicationId: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The specified OAuthApplicationId is invalid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the OAuthApplicationId.</td>
</tr>
<tr>
<td>222207712</td>
<td>Missing 'name' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The name parameter for the OAuthApplication is required, but none was specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add a value for name, that is, the name of the OAuthApplication. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207713</td>
<td>Missing 'dataStore' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The dataStore parameter is required, but none was specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add a value for dataStore, that is, the name of the dataStore. The dataStore ID can be obtained from the <code>&lt;base&gt;/datastores</code> resource.</td>
</tr>
<tr>
<td>222207714</td>
<td>Missing 'oauthAppId' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The oauthAppId parameter is required, but none was specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add a value for oauthAppId. This property is generated by Hybrid Data Pipeline and cannot be changed once assigned. The ID is used to identify the data source type in data source references.</td>
</tr>
<tr>
<td>222207715</td>
<td>Missing 'refreshToken' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The refreshToken was not specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the refreshToken specified in in the payload.</td>
</tr>
<tr>
<td>222207716</td>
<td>Missing 'clientId' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The clientId parameter is not in the payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> The clientId parameter is required. Visit the Google Developers Console to obtain OAuth 2.0 credentials that are known to both Google and your application.</td>
</tr>
<tr>
<td>222207717</td>
<td>Missing 'clientSecret' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The clientSecret parameter is not in the payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> The clientSecret parameter is required. Visit the Google Developers Console to obtain OAuth 2.0 credentials that are known to both Google and your application.</td>
</tr>
<tr>
<td>222207718</td>
<td>Problem validating the OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207719</td>
<td>OAuthProfile name must be unique for a given OAuthApplication.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> OAuthProfile name must be unique for a given OAuthApplication.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Use a different OAuthProfile name.</td>
</tr>
<tr>
<td>222207720</td>
<td>That OAuthApplication Name is invalid. Please choose another name.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The specified OAuthApplication Name is invalid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Choose another name. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>222207721</td>
<td>You cannot change the DataStore of a OAuthApplication.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The dataStore value cannot be changed.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Create a new OAuthApplication for the data store, that is, the data source type, that you want to use.</td>
</tr>
</tbody>
</table>
Chapter 8: Querying data stores with SQL

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207722</td>
<td>Problem getting the OAuthProfile Statistics at this time. Please try again at another time.</td>
</tr>
</tbody>
</table>
| 222207723  | DataStore {0} does not support OAuth.  
**Cause:** The `dataStore` parameter specified a data store that does not support OAuth.  
**Action:** Check with your database administrator. |

**HTTP Response Codes Returned by the Hybrid Data Pipeline Management Data Sources API**

Hybrid Data Pipeline Management Data Sources API returns standard HTTP response codes as described in the following table, under the conditions listed in the description. The descriptions differ somewhat from the general description found earlier in this document.

**Table 208: HTTP Error Messages for the Data Sources API**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 200        | OK  
The request was successfully completed. If this request created a new resource that is addressable with a URI, and a response body is returned containing a representation of the new resource, a 200 status will be returned with a location header containing the canonical URI for the newly created resource. |
| 201        | Created  
A request that created a new resource was completed and no response body containing a representation of the new resource is being returned. A location header containing the canonical URI for the newly created resource will be returned. |
| 400        | Bad Request  
The JSON request is invalid. |
| 401        | Not Authorized  
The user is not authorized. An invalid user name and/or password was used. |
| 404        | Not Found  
The `<DataSource>` was not found, where `<resource_type>` is `DataSource`. |
| 500        | Internal Server Error  
The server encountered an unexpected condition which prevented it from fulfilling the request. |
| 501        | Not Implemented  
The server currently does not support the functionality required to fulfill the request. |
Data Sources API error messages

This section describes error messages you may receive from the Data Sources API. Each error message is followed by a possible cause and recommended actions, if applicable.

Table 209: Error messages for the Data Sources API

<table>
<thead>
<tr>
<th>Error code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207000</td>
<td>Problem updating your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207001</td>
<td>Problem retrieving your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207002</td>
<td>Invalid DataSource Option: {0}.</td>
</tr>
<tr>
<td>222207003</td>
<td>There is a problem connecting to the DataSource. {0}.</td>
</tr>
<tr>
<td>222207004</td>
<td>There is no DataSource with that id: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The DataSource ID is incorrect. The data source ID may have been entered incorrectly, or the data source ID might have been invalidated by the administrator.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Correct the DataSource ID.</td>
</tr>
<tr>
<td>222207005</td>
<td>Expected values for connectType: 'Cloud' / 'Hybrid'. Your value was {0}. Please try again with proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The connectionType parameter specified a value other than Cloud or Hybrid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify a valid value.</td>
</tr>
<tr>
<td>222207006</td>
<td>Problem deleting your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The DataSource couldn't be deleted at this time.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try again later.</td>
</tr>
<tr>
<td>222207007</td>
<td>Invalid JSON Input: {0}</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The JSON input was not valid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Correct the JSON statement and retry the query.</td>
</tr>
<tr>
<td>222207008</td>
<td>connectionType is not allowed to be changed. It must remain : {0}.</td>
</tr>
<tr>
<td>222207009</td>
<td>Expected values for map:'refresh'/recreate'/none'. Your value was {0}. Please try again with proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The map parameter specified an invalid value.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Change the value for the map parameter. The valid values are refresh, recreate, and none.</td>
</tr>
<tr>
<td>222207010</td>
<td>Missing 'connectionType' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The connectionType parameter is missing, or no value was defined.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the payload. Add the connectionType and a valid value.</td>
</tr>
</tbody>
</table>
## Error codes

<table>
<thead>
<tr>
<th>Error code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 222207011           | Invalid DataSource ID: {0}.  
**Cause**: The specified DataSource ID is invalid.  
**Action**: Check the DataSource ID. |
| 222207012           | You are not authorized to create a DataSource with this DataStore id: {0}. Please contact Technical Support if you would like to upgrade your account.  
**Cause**: The DataStore you specified is not included in your subscription plan, or you are not authorized to use the DataStore. For example, the Hybrid Data Pipeline administrator might have limited the number of users who can access Salesforce.  
**Action**: Contact your Hybrid Data Pipeline administrator or Technical Support. |
| 222207013           | Problem validating your DataSource at this time. Please try again at another time.  
**Cause**: There was a problem validating your DataSource.  
**Action**: Try validating your DataSource later. |
| 222207014           | You already have a DataSource with the name {0}. Please choose another name.  
**Cause**: A data source with that name already exists.  
**Action**: Choose a different name for the data source. |
| 222207015           | Invalid DataStore ID: {0}.  
**Cause**: The DataStoreID specified is not valid.  
**Action**: Check the DataStoreID specified in in the payload. You can get the DataSourceID from the DataStores resource. |
| 222207016           | Missing 'name' in payload.  
**Cause**: The `name` parameter is not in the payload.  
**Action**: The `name` parameter is required. The name must contain only alphabetic characters and the underscore, and must begin with a letter. |
| 222207017           | Problem refreshing your DataSource at this time. Please try again at another time.  
**Cause**: The DataSource could not be refreshed.  
**Action**: Try refreshing the DataSource later. |
| 222207018           | {0} is an unrecognized argument for /map. Expected 'map' and/or 'model' only.  
**Cause**: An unrecognized argument was used for `map`.  
**Action**: The only valid arguments are `map` and `model`. |
| 222207019           | Missing 'id' in payload.  
**Cause**: The `id` property is the data source id used to reference the data source in the Hybrid Data Pipeline Management API URLs.  
**Action**: Add the data source id for the data source. |
<table>
<thead>
<tr>
<th>Error code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207020</td>
<td>Missing 'password' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The password property is missing.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the payload and add a valid password.</td>
</tr>
<tr>
<td>222207021</td>
<td>DataStore is not allowed to be changed. It must remain: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The DataStore value cannot be changed.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the JSON string.</td>
</tr>
<tr>
<td>222207022</td>
<td>There was a problem deleting the DataSource. Multiple rows were somehow deleted. {0}</td>
</tr>
<tr>
<td>222207023</td>
<td>Problem connecting to your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem connecting to the data source.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try connecting later.</td>
</tr>
<tr>
<td>222207024</td>
<td>Problem retrieving your DataSources at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem retrieving your data sources.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207025</td>
<td>Problem creating your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem creating your data sources.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207026</td>
<td>Missing 'dataStore' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The payload did not specify a valid dataStore element.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Add the dataStore to the payload.</td>
</tr>
<tr>
<td>222207027</td>
<td>There is a problem getting the DataStore(s) at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem getting your data sources.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207028</td>
<td>Missing 'userId' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The userId parameter was not in the payload, or no value was defined.</td>
</tr>
</tbody>
</table>
|             | **Action**: Make sure the payload contains the userId parameter with a valid user name.
<table>
<thead>
<tr>
<th>Error code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207029</td>
<td>Expected values for model: 'refresh' / 'none'. Your value was {0}. Please try again with proper value.</td>
</tr>
<tr>
<td><strong>Cause:</strong></td>
<td>The model parameter specified an invalid parameter.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Check the model parameter and change the value. The valid values are refresh and none.</td>
</tr>
<tr>
<td>222207030</td>
<td>Data Source 'id' in the JSON Request must match the resource. ie. /datasources/&lt;id&gt;. DataSource 'id' is an optional field.</td>
</tr>
<tr>
<td><strong>Cause:</strong></td>
<td>The data source ID is generated by Hybrid Data Pipeline and cannot be changed.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Including the data source ID in the JSON request is optional. When the ID is included in the JSON request, make sure it matches the resource.</td>
</tr>
<tr>
<td>222207031</td>
<td>Invalid userName {0}.</td>
</tr>
<tr>
<td><strong>Cause:</strong></td>
<td>The specified user name is not valid.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Enter a valid user name.</td>
</tr>
<tr>
<td>222207032</td>
<td>Must supply 'map' and/or 'model' in your payload.</td>
</tr>
<tr>
<td><strong>Cause:</strong></td>
<td>Either map or model must be specified in the payload.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Add map or model to the payload.</td>
</tr>
<tr>
<td>222207033</td>
<td>Problem retrieving the members of your DataSource Group. Please try again at another time.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Try again later.</td>
</tr>
<tr>
<td>222207034</td>
<td>Problem updating the members of your DataSource Group. Please try again at another time.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Try again later.</td>
</tr>
<tr>
<td>222207035</td>
<td>Problem creating one or more new member DataSources for your DataSource Group. Please try again at another time.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Try again later.</td>
</tr>
<tr>
<td>222207036</td>
<td>Problem removing one or more member DataSource from your DataSource Group. Please try again at another time.</td>
</tr>
<tr>
<td><strong>Cause:</strong></td>
<td>A problem occurred when attempting to remove one or more member data sources from your data source group.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Try removing the member data sources from the data source group later.</td>
</tr>
<tr>
<td>222207037</td>
<td>Only DataSource Groups can have member DataSources assigned.</td>
</tr>
<tr>
<td><strong>Cause:</strong></td>
<td>An attempt was made to add a member data source to a data source that was not defined as a data source group.</td>
</tr>
<tr>
<td><strong>Action:</strong></td>
<td>Add the member DataSource to a data source group.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207038</td>
<td>DataSource {0} must be a DataSource Group when used in this way.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> An attempt was made to use a simple or member data source as a data source group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> You can't change the data source into being a data source group. Specify a data source that is a data source group for this action.</td>
</tr>
<tr>
<td>222207039</td>
<td>DataSource {0} cannot be a DataSource Group when used in this way.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> An attempt was made to use a data source group when a simple or member DataSource was needed.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Use a simple data source or a member data source.</td>
</tr>
<tr>
<td>222207040</td>
<td>An existing DataSource {0} was seen while adding new DataSource members to a DataSource Group.</td>
</tr>
<tr>
<td>222207041</td>
<td>The DataSource cannot be removed because it is used in one or more DataSource Groups: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> An attempt was made to delete a data source that is a member of one or more data source group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Remove the data source from each data source group that it is a member of.</td>
</tr>
<tr>
<td>222207042</td>
<td>When updating a DataSource Group, a &quot;members&quot; section must be supplied.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> An attempt was made to update a data source group, but the payload did not contain a members parameter.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add a members parameter to the options object.</td>
</tr>
<tr>
<td>222207043</td>
<td>You are not authorized to update a {0} DataSource (DataStore id: {1}). Please contact Customer Support if you would like to upgrade your account.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> You are not authorized to update the specified data source for the data source type.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check with your Hybrid Data Pipeline administrator to see if the authorization can be changed. For example, the subscription might be configured for 5 users to update Salesforce.</td>
</tr>
<tr>
<td>222207044</td>
<td>A DataSource Group connectionType must be 'Group'. Your value was {0}. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The value specified for connectionType was invalid for a data source group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Change the value of connectionType to Group.</td>
</tr>
<tr>
<td>222207045</td>
<td>MaximumEntityNameLength must be an integer between 10 and 128 inclusive, but your value was {0}. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The value specified for MaximumEntityNameLength was not an integer between 10 and 128 inclusive.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify an integer between 10 and 128 inclusive.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207046</td>
<td>MaximumEntityNameLength is outside the valid range of 10 to 128 inclusive. but your value was {0}. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The value specified for MaximumEntityNameLength was not in the valid range.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify an integer between 10 and 128, inclusive.</td>
</tr>
<tr>
<td>222207047</td>
<td>The entity prefix for member datasources must be specified. For source {0}, it was not. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> Each member data source must specify a unique entity prefix.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify a unique entity prefix that is less than half the length of the value specified for MaximumEntityNameLength.</td>
</tr>
<tr>
<td>222207048</td>
<td>The entity prefix for source {0} must be less than half the maximum entity name length. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The entity prefix for the specified data source must specify a unique entity prefix that is less than half the maximum entity name length.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify a unique entity prefix that is less than half the length of the value specified for MaximumEntityNameLength.</td>
</tr>
<tr>
<td>222207049</td>
<td>All of the entity prefixes within a DataSource Group must be unique. DataSource {0} has a duplicate. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> Each member data source must specify a unique entity prefix.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify a unique entity prefix that is less than half the length of the value specified for MaximumEntityNameLength.</td>
</tr>
<tr>
<td>222207050</td>
<td>Entity prefixes cannot contain underscores, but DataSource {0} has one. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The entity prefix can contain only alphanumeric characters and can't contain an underscore.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Modify the entity prefix.</td>
</tr>
<tr>
<td>222207051</td>
<td>The entity prefix name for member DataSource {0} does not follow OData guidelines. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The entity prefix can contain only alphanumeric characters and must begin with an alphabetic character.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Correct the entity prefix.</td>
</tr>
<tr>
<td>222207052</td>
<td>Problem getting the status of your OData Model Creation. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when getting the status of the OData Model Creation.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try again later.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207053</td>
<td>Problem starting creation of your OData Model. Please try again at another time.</td>
</tr>
<tr>
<td>222207054</td>
<td>Cannot start the OData Model Creation because it is currently running. Please see the documentation if you wish to restart the creation.</td>
</tr>
<tr>
<td>222207055</td>
<td>The status was changed during the process of the request. Please verify and send request again if needed.</td>
</tr>
<tr>
<td>222207056</td>
<td>You cannot create an OData Model for a DataSource Group.</td>
</tr>
<tr>
<td>222207057</td>
<td>You cannot refresh/recreate the map of a DataSource Group.</td>
</tr>
<tr>
<td>222207058</td>
<td>DataSource {0} must have an OData map.</td>
</tr>
<tr>
<td>222207059</td>
<td>Test connect cannot be performed on a DataSource Group. To test connectivity, the member data sources of the group should be tested.</td>
</tr>
<tr>
<td>222207060</td>
<td>There are duplicate members in the payload. Please remove the duplicates and try again.</td>
</tr>
<tr>
<td>222207061</td>
<td>Member {0} already exists in the DataSource Group that matches one in your payload; please adjust your payload and try again.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207062</td>
<td>The schema ({0}) does not exist.&lt;br&gt;&lt;br&gt;&lt;b&gt;Cause:&lt;/b&gt; The specified schema does not exist.&lt;br&gt;&lt;br&gt;&lt;b&gt;Action:&lt;/b&gt; Check the schema name. If necessary, use the Get Schemas API for a list of valid schemas.</td>
</tr>
<tr>
<td>222207063</td>
<td>The table ({0}) does not exist under schema ({1}).&lt;br&gt;&lt;br&gt;&lt;b&gt;Cause:&lt;/b&gt; The specified table does not exist under the specified schema.&lt;br&gt;&lt;br&gt;&lt;b&gt;Action:&lt;/b&gt; Check the table name and schema name.</td>
</tr>
<tr>
<td>222207064</td>
<td>Problem retrieving the schemas at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207065</td>
<td>Problem retrieving the tables at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207066</td>
<td>Problem retrieving the columns at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207067</td>
<td>Problem retrieving the primary keys at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207068</td>
<td>Problem retrieving the table details at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207069</td>
<td>Invalid OAuthProfileID: {0}.&lt;br&gt;&lt;br&gt;&lt;b&gt;Cause:&lt;/b&gt; The specified OAuthProfileID is not valid.&lt;br&gt;&lt;br&gt;&lt;b&gt;Action:&lt;/b&gt; Correct the OAuthProfileID.</td>
</tr>
<tr>
<td>222207070</td>
<td>The OAuthProfile data store ({0}) does not match the DataSource data store({1}).&lt;br&gt;&lt;br&gt;&lt;b&gt;Cause:&lt;/b&gt; The specified OAuthProfile data source type does not match the data source type specified in the DataSource.&lt;br&gt;&lt;br&gt;&lt;b&gt;Action:&lt;/b&gt; Check the OAuthProfile data source type and the DataSource data source type.</td>
</tr>
</tbody>
</table>

**HTTP Response Codes Returned by the Hybrid Data Pipeline Management Data Sources API**

Hybrid Data Pipeline Management Data Sources API returns standard HTTP response codes as described in the following table, under the conditions listed in the description. The descriptions differ somewhat from the general description found earlier in this document.

**Table 210: HTTP Error Messages for the Data Sources API**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK&lt;br&gt;The request was successfully completed. If this request created a new resource that is addressable with a URI, and a response body is returned containing a representation of the new resource, a 200 status will be returned with a location header containing the canonical URI for the newly created resource.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
</tr>
<tr>
<td></td>
<td>A request that created a new resource was completed and no response body containing a representation of the new resource is being returned. A location header containing the canonical URI for the newly created resource will be returned.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
</tr>
<tr>
<td></td>
<td>The JSON request is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Not Authorized</td>
</tr>
<tr>
<td></td>
<td>The user is not authorized. An invalid user name and/or password was used.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
</tr>
<tr>
<td></td>
<td>The &lt;DataSource&gt; was not found, where &lt;resource_type&gt; is DataSource.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td></td>
<td>The server encountered an unexpected condition which prevented it from fulfilling the request.</td>
</tr>
<tr>
<td>501</td>
<td>Not Implemented</td>
</tr>
<tr>
<td></td>
<td>The server currently does not support the functionality required to fulfill the request.</td>
</tr>
</tbody>
</table>

**Performance tuning**

For some data stores, you can tune the queries for better performance.

**Oracle Marketing Cloud bulk operations**

Hybrid Data Pipeline supports Oracle Eloqua bulk operations with some limitations. The Enable Bulk Load connection option can be used to enable or disable Oracle Eloqua bulk operations.

*Note:* Bulk operations are most efficient for queries that return large amounts of data for a relatively small set of columns. For example, `SELECT folderid, name, country, c_website FROM Account WHERE country='Switzerland'` has only four columns but many rows.

Bulk operations for some Select queries with a Top \( n \) clause are not supported because it is usually faster to use a standard query to fetch more columns for a few rows than to use a bulk operation. Nevertheless, you can use the Bulk Top Threshold connection option to control, in part, how queries with a Top \( n \) clause are handled. When bulk operations are enabled, bulk load is used to process queries with a Top \( n \) clause. The default value of Bulk Top Threshold is 1000.

*Note:* Bulk load must be enabled to support queries against Activity objects. If bulk load is not enabled, queries against Activity objects will fail.
Queries on Account and Contact tables have additional limitations. The following criteria must be met for queries on Account and Contact tables.

- The result must have multiple rows.
- The result cannot have more than 250 columns.
- The result must include at least one user-defined column.
- The query must either have no TOP \( n \) clause, or the value of \( n \) in the TOP \( n \) clause must be greater than the value specified in the Bulk Top Threshold option.
- The query can only include columns that the bulk interface supports. For more information, see the table below.

**Table 211: Columns that cannot be retrieved using bulk operations**

<table>
<thead>
<tr>
<th>Account table</th>
<th>Contact table</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessedAt</td>
<td>accessedAt</td>
</tr>
<tr>
<td>createdBy</td>
<td>bouncebackDate</td>
</tr>
<tr>
<td>currentStatus</td>
<td>currentStatus</td>
</tr>
<tr>
<td>Description</td>
<td>createdBy</td>
</tr>
<tr>
<td>folderId</td>
<td>Description</td>
</tr>
<tr>
<td>Permissions</td>
<td>folderId</td>
</tr>
<tr>
<td>scheduledFor</td>
<td>Permissions</td>
</tr>
<tr>
<td>sourceTemplateId</td>
<td>scheduledFor</td>
</tr>
<tr>
<td>updatedBy</td>
<td>sourceTemplateId</td>
</tr>
<tr>
<td></td>
<td>subscriptionDate</td>
</tr>
<tr>
<td></td>
<td>updatedBy</td>
</tr>
<tr>
<td></td>
<td>unsubscriptionDate</td>
</tr>
</tbody>
</table>

The following table provides some query examples and describes why they would not take advantage of bulk operations and offers suggestions for modifying them. However, there will obviously be use cases where an application will use queries that cannot be returned using bulk operations.

<table>
<thead>
<tr>
<th>Query</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT * FROM Contact WHERE Country='Switzerland'</td>
<td>There are more than 100 columns in the Contact table. To take advantage of bulk operations, constrain the SELECT statement to a set of 100 columns or less.</td>
</tr>
<tr>
<td>SELECT * FROM ContactList WHERE Region='East'</td>
<td>The ContactList table is not supported for bulk operations.</td>
</tr>
<tr>
<td>Query</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SELECT Id, C_Website FROM Account WHERE Id=17</td>
<td>This returns one row.</td>
</tr>
<tr>
<td>SELECT Id, C_Website FROM Contact WHERE Country='Switzerland' AND Region='EAST'</td>
<td>More than one criterion or comparison operators are used in the WHERE clause.</td>
</tr>
<tr>
<td></td>
<td><strong>Tips:</strong></td>
</tr>
<tr>
<td></td>
<td>• If a query has zero or one comparison operators (meaning one of =, &lt;=, &lt;, &gt;, &gt;=, &lt;&gt;) in the WHERE clause, the query will usually be processed using the bulk operations.</td>
</tr>
<tr>
<td></td>
<td>• If a query contains the LIKE operator in the WHERE clause, the query is not processed using the bulk operations.</td>
</tr>
</tbody>
</table>

### Efficient queries

Hybrid Data Pipeline supports queries based on the definition of Oracle Eloqua minimal, partial, and complete column sets. If you know which type of column you are querying, you can optimize queries by following these guidelines.

**Note:** Refer to your Oracle Eloqua documentation for details on minimal, partial, and complete column sets.

- A minimal column set provides the best performance. For example: `SELECT name, description, createdBy FROM Account WHERE scheduledFor='May'`

- A partial column set provides the next best performance. For example, `SELECT name, description, createdBy, country FROM Account WHERE scheduledFor='May'` is processed faster than: `SELECT name, description, createdBy, country, c_website FROM Account WHERE scheduledFor='May'`, which contains columns from Complete Column Set.

- A complete column set is processed faster than a query that uses the bulk interface only when the required number of records is less than the actual number of records that the bulk query would return. For example, the following query would return all rows: `SELECT name, country, c_website FROM Account WHERE scheduledFor='May'`. While Hybrid Data Pipeline would attempt to use bulk operations for such a query, suppose you were only interested in the first 500 of 10,000 records. In that case, the query `SELECT TOP 500 name, country, c_website FROM Account WHERE scheduledFor='May'` would probably be faster, even though it would not be fetched in a bulk operation.
Configuring an On-Premises Connector

For details, see the following topics:

• Configuring the On-Premises Connector

Configuring the On-Premises Connector

You can use the On-Premises Configuration Tool to change settings you established during the installation process. You may use the Configuration Tool to change the user id and password used to register the On-Premises Connector with Hybrid Data Pipeline, and to change the label used to identify the Connector in the configuration dialogs. The Configuration Tool also allows you to see the Hybrid Data Pipeline Connector ID that is used to register the On-Premises Connector with Hybrid Data Pipeline.

1. Start the On-Premises Configuration Tool by selecting Windows Start Menu > All Programs > Progress DataDirect Hybrid Data Pipeline On-Premises Connector. Then select Configuration Tool.
2. Enter a name for your On-Premises Connector instance in the Connector Label field.
3. Enter your Hybrid Data Pipeline user id and password in their corresponding fields.
4. Click Save. This registers the On-Premises Connector to the Hybrid Data Pipeline service.
5. Select the Status tab and click Test to verify that the On-Premises Connector configuration is correct. All tests should all have a green check mark, showing the test was successful.
Restarting the On-Premises Connector

You must restart the On-Premises Connector whenever any configuration changes are made using the Configuration Tool. Follow these steps to start and restart On-Premises Connector services.

1. Select **Windows Start Menu > All Programs > Progress DataDirect Hybrid Data Pipeline On-Premises Connector**, and select **Stop Services**.
2. After the service is stopped, return to the **Progress DataDirect Hybrid Data Pipeline On-Premises Connector** program group and select **Start Services**.
3. From the **Progress DataDirect Hybrid Data Pipeline On-Premises Connector** program group, select the **Configuration Tool**.
4. Select the Status tab and click **Test** to verify that the On-Premises connector configuration is correct.

Each test should have a green check mark, showing the test was successful. If a red x appears next to any tests, you should reenter the information or see Troubleshooting the On-Premises Connector on page 1036 to troubleshoot the issue.

Determining the Connector information

The On-Premises Configuration Tool allows you to see the Hybrid Data Pipeline Connector ID being used to register the On-Premises Connector with Hybrid Data Pipeline. You may also use the Configuration Tool to change the User ID and Password used to register the On-Premises Connector with Hybrid Data Pipeline, and to change the label used to identify the Connector in the configuration dialogs.

When you configure a Hybrid Data Pipeline data source to connect to an on-premises Data Store such as OpenEdge using the On-Premises Connector, you must select the Connector from a drop-down list.

**Note:** By default, only the owner of the On-Premises Connector can use the Connector to access data sources behind the firewall. The owner of the On-Premises Connector can grant other Hybrid Data Pipeline users permission to use the Connector. The User ID of the owner of the On-Premises Connector is shown in the User ID field of the General tab in the Configuration Tool. The Hybrid Data Pipeline Management API provides a set of REST calls that allow the owner to manage the list of users that can access the On-Premises Connector. Through the APIs, the owner can add and remove Hybrid Data Pipeline users to the list of users that can use the Connector. See the "Getting Started with Hybrid Data Pipeline" in the user's guide for more information.

1. Select **Configuration Tool** in the Hybrid Data Pipeline On-Premises Connector program group. Alternatively, navigate to the directory **install_dir\OPDH\config** and double-click the **opconfig.bat** file, or type **opconfig** from a command prompt.

The General tab of the Hybrid Data Pipeline On-Premises Connector Configuration Tool displays the Connector ID and the User ID used when installing the On-Premises Connector.
2. Select the Connector ID string and copy it to a text file that you can refer to when you use the Hybrid Data Pipeline Management APIs.

3. If you want to change the label, which by default is the name of the computer, enter a unique descriptive name in the Connector Label field. The maximum length is 255 characters. This label appears in the Connector ID drop-down list on the configuration dialogs.
   If you have already used the label on another Connector, you are prompted to enter a different label. For example, you might change Production to Production(West).

4. If you want to change the User ID and Password that was used to register the On-Premises Connector, enter a valid Progress Id and password in the User ID and Password fields.
   If the User ID and Password are not valid, a message is returned.

5. Click Save to persist your settings, as well as changes on other Configuration Tool tabs. If you save your settings, then close and reopen the Configuration Tool, the saved settings are displayed automatically.

6. Click Close to exit the Configuration Tool.

   **Note:** If you uninstall the On-Premises Connector and later re-install it, the Connector ID changes, even if you reuse the Connector label. In this case, you must update any data sources created with the original Connector ID. For each data source, select the label for the newer Connector. If you shared the Connector with other users, make sure that they update their data sources.

### Defining the proxy server

The On-Premises Connector must communicate with the Hybrid Data Pipeline service using the Internet. If your network environment requires a proxy to access the public internet, you provide the proxy host name and port on the Proxy tab of the Configuration Tool and specify what type of proxy authentication to use. You might need to contact your network administrator to determine what proxy information you need to provide.
1. Open the Configuration Tool, and click the **Proxy** tab. If you provided the proxy connection information when you installed the Connector, the fields are automatically populated with that information.

![Configuration Tool Proxy tab](image)

2. Select the type of proxy authentication needed in your environment:
   - Select **No Proxy Authentication** if the proxy server does not require authentication.
   - Select **HTTP Proxy Authentication** if the proxy server requires that all requests be authenticated using the HTTP Basic authentication protocol.
   - Select **NTLM Proxy Authentication** if the proxy server requires that all requests be authenticated using the NTLM authentication protocol.

3. Provide the connection information for the proxy server. You may need to contact your network administrator for the proxy host name and port number, and if required, the proxy user name and password.
   - **Proxy Host** specifies the Host name and, optionally, the domain of the proxy server. The value can be a host name, a fully qualified domain name, or an IPv4 or IPv6 address.
   - **Proxy Port** specifies port number where the proxy server is listening.
   - **Proxy User** specifies the user name needed to connect to the proxy server, if HTTP or NTLM authentication is specified. If NTLM authentication is specified, the user name must be in the form `domain\user`.
   - **Proxy Password** specifies the password needed to connect to the proxy server, if you are using HTTP Basic or NTLM authentication.

4. Click **Save** to persist your Proxy settings, as well as changes you made on other Configuration Tool tabs. If you save your settings, then close and reopen the Configuration tool, the saved settings are automatically repopulated.

5. Click **Close** to exit the Configuration Tool.

**Configuring On-Premises Connector memory resources**

In most cases, the default memory allocated to the On-Premises Connector is sufficient, allowing for a small number of open connections and simultaneous requests. However, depending on the number and complexity of concurrent requests in your environment, you might need to increase the memory allocated to the On-Premises Connector, the number of concurrent request it can process, or both, to handle the query volume. If the memory allocation and concurrent request settings are at the high end of the range, you might want to consider using multiple On-Premises Connectors and configure load balancing to share the load between the Connectors.
1. Open the Configuration Tool, and click the **Resource** tab.

2. Select a preset memory load from the dropdown list, or specify custom values. The default resource settings are sufficient for most On-Premises Connector installations, allowing for a small number of open connections and simultaneous requests. Note that because the values for the High and Very High settings exceed the limits of a 32-bit Windows platform, they are not available when using the On-Premises Connector on a 32-bit inmachine.

   **Min Memory Size (MB)** specifies the minimum number of megabytes used by the On-Premises Connector's JVM. It must be less than or equal to the Max Memory Size. The valid range is 128 to 16384.

   **Max Memory Size (MB)** specifies the maximum number of megabytes used by the On-Premises Connector's JVM. Be sure that your system has at least this much memory available for use by the Connector. The valid range is 256 to 16384.

   **Concurrent Requests** specifies the maximum number of concurrent requests, such as login and execute, that are supported. The valid range is 50 to 1000.

<table>
<thead>
<tr>
<th>Memory Load</th>
<th>Minimum Memory Size (MB)</th>
<th>Maximum Memory Size (MB)</th>
<th>Concurrent Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>128</td>
<td>256</td>
<td>200</td>
</tr>
<tr>
<td>Moderate</td>
<td>2048</td>
<td>2048</td>
<td>500</td>
</tr>
<tr>
<td>High</td>
<td>4096</td>
<td>4096</td>
<td>800</td>
</tr>
<tr>
<td>Very High</td>
<td>8192</td>
<td>8192</td>
<td>1000</td>
</tr>
<tr>
<td>Custom</td>
<td>8192</td>
<td>8192</td>
<td>1000</td>
</tr>
</tbody>
</table>

3. Click **Save** to persist your settings on this and other Configuration Tool tabs. If you save your settings, then close and reopen the Configuration tool, the saved settings are displayed automatically.

4. Click **Close** to exit the Configuration Tool.
Determining the version

The Version tab shows the versions of the Hybrid Data Pipeline connectivity service and components of the On-Premises Connector.

1. Open the Configuration Tool, and click the Version tab.

2. When requested, provide this information to Progress Hybrid Data Pipeline technical support.

3. Click Save to persist your settings on other Configuration Tool tabs. If you save your settings, then close and re-open the Configuration tool, the saved settings are displayed automatically.

4. Click Close to exit the Configuration Tool.

Checking the configuration status

Use the Status tab of the On-Premises Connector Configuration Tool to determine whether the On-Premises Connector is configured correctly. When you click Test, connections are made to the different services used by the On-Premises connector. (Because the proxy password value is encrypted when added to the Configuration Tool, you are prompted to re-enter your Proxy Password when you click Test.) The On-Premises Connector is configured properly if a green check is shown next to each service. Click Details for additional status information. If a red x is shown next to any service, see the table in Troubleshooting the On-Premises Connector on page 1036 or contact Progress Technical Support.

Configuring failover and balancing requests across multiple On-Premises Connectors

Hybrid Data Pipeline supports failover and balancing the load of requests across multiple On-Premises Connectors.

You can use the Hybrid Data Pipeline Connector API to configure failover across multiple On-Premises Connectors. If a request to a specific On-Premises Connector fails and the connectors are configured for failover, the failed request will be retried on another On-Premises Connector.

You can also use the Connector API to balance the load of requests across multiple On-Premises Connectors. This allows more traffic to be directed to a specific connector if needed. For example, if Connector1 is running on a faster server than Connector2, a higher number of requests can be sent to Connector1.
Configuring the Microsoft Dynamics CRM On-Premises data source for Kerberos

During installation of the On-Premises Connector, the files required for Kerberos authentication are installed in the \%jre\%lib\%security subdirectory of your product installation directory:

- krb5.conf is a Kerberos configuration file containing values for the Kerberos realm and the KDC name for that realm. You must modify the generic file that is installed for your environment.
- JDBCDriverLogin.conf file is a configuration file that specifies which Java Authentication and Authorization Service (JAAS) login module to use for Kerberos authentication. This file loads automatically unless the java.security.auth.login.config system property is set to load another login configuration file. You can edit this file, but the On-Premises Connector must be able to find the JDBC_DRIVER_01 entry to configure the JAAS login module. Refer to your J2SE documentation for information about setting options in this file.

**Note:** You must download the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files for JDKJRE 7 at http://www.oracle.com/technetwork/java/javase/downloads/index.html. Unzip the files into the \%jre\%lib\%security subdirectory of your product installation directory.

To configure the On-Premises Connector for Microsoft Dynamics CRM:

1. Set the AuthenticationMethod property to kerberos.
2. Modify the krb5.conf file to contain your Kerberos realm name and the KDC name for that Kerberos realm by editing the file with a text editor. Alternatively, you can specifying the system properties, java.security.krb5.realm and java.security.krb5.kdc. You may need to contact your network administrator for the Kerberos realm name and KDC name.

**Note:** If using Windows Active Directory, the Kerberos realm name is the Windows domain name and the KDC name is the Windows domain controller name.

For example, if your Kerberos realm name is XYZ.COM and your KDC name is kdc1, your krb5.conf file would look like this:

```
[libdefaults]
default_realm = XYZ.COM

[realms]
XYZ.COM = {
kdc = kdc1
}
```

If the krb5.conf file does not contain a valid Kerberos realm and KDC name, the following exception is thrown:

```
Message: [DataDirect][JDBC Cloud Driver][Microsoft Dynamics CRM]Could not establish a connection using integrated security: No valid credentials provided
```

The krb5.conf file loads automatically unless the java.security.krb5.conf system property is set to load another Kerberos configuration file.
Troubleshooting the On-Premises Connector

Use the Status tab of the On-Premises Connector Configuration Tool to determine whether the On-Premises Connector is configured correctly. When you click Test, connections are made to the different services used by the On-Premises Connector. The On-Premises Connector is configured properly if a green check is shown next to each service. If a red x is shown next to any service, see the troubleshooting table below or contact Progress Technical Support. Click Details for additional status information.

The following table can be used to help troubleshoot configuration properties. If a red x is shown next to a service, see the recommendations for that service for possible actions to correct the problem. Then, click Test again. If the recommended actions do not correct the problem, contact Progress Technical Support.

If changes were made to correct any configuration problems, click Save to save the changes, and then click Test to recheck the status.

<table>
<thead>
<tr>
<th>Service</th>
<th>Recommended Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud Service</td>
<td>Does your network environment require a Proxy? If so, verify that the Proxy connection information is specified correctly on the Proxy Tab of the Configuration Tool.</td>
</tr>
<tr>
<td>Notification Service</td>
<td>Is the UserId and Password for the On-Premises Connector correct? The user name and password should be your Progress ID and password. You can change the connector user id and password in the On-Premises Connector Configuration Tool.</td>
</tr>
<tr>
<td>On-Premise Access Service</td>
<td>Does your network environment require a proxy? If so, verify that the Proxy connection information is specified correctly on the Proxy Tab of the Configuration Tool.</td>
</tr>
<tr>
<td>Connector Service</td>
<td>Are the On-Premises Connector services running on this client machine? The On-Premises Connector services can be started by selecting the Start Services item under the Progress DataDirect Hybrid Data Pipeline On-Premises Connector folder in the Start Menu.</td>
</tr>
</tbody>
</table>
Hybrid Data Pipeline API reference

Hybrid Data Pipeline provides a representational state transfer (REST) application programming interface (API) for managing Hybrid Data Pipeline connectivity service resources.

Hybrid Data Pipeline APIs use HTTP Basic Authentication to authenticate user accounts. The Hybrid Data Pipeline user ID and password are encoded in the Authorization header. The Hybrid Data Pipeline user specified in the Authorization header is the authenticated user.

**Note:** Administrators can execute a number of API operations on behalf of standard Hybrid Data Pipeline users. For details, see Managing resources on behalf of users on page 1281 and User provisioning on page 107.

To execute REST calls, you must pass a valid REST URL and pass a valid username and password to authenticate with basic authentication. A REST URL must include a base and resource-specific information. The base includes the Web protocol, a server name, and a port number, while resource-specific information provides a path to a particular resource necessary for performing an API operation. For example:

```plaintext
https://MyServer:8443/api/mgmt/datasources
```

**Note:** The port number is only required if the Hybrid Data Pipeline server or load balancer is configured to use a port other than 443 for SSL or 80 for non-SSL connections.

The syntax for a REST URL can be described as follows.

```plaintext
webprotocol://servername:portnumber/resourceinfo
```

where

```plaintext
webprotocol
```

is the Web protocol, such as HTTP or HTTPS, used to connect to your Hybrid Data Pipeline instance.
servername

is the name of the machine hosting the Hybrid Data Pipeline service, or the name of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service.

portnumber

is the port number of the machine hosting the Hybrid Data Pipeline service, or the port number of the machine hosting the load balancer used to route requests to the Hybrid Data Pipeline service. For a standalone installation, the port number is specified as the Server Access Port during installation. For a load balancer installation, the port number must be either 80 for http or 443 for https. Whenever port 80 or 433 are used, it is not necessary to include the port number in the URL.

resourceinfo

is resource-specific information that provides a path to a particular Hybrid Data Pipeline resource necessary to perform an API operation.

Compatibility Note

Future versions of Hybrid Data Pipeline APIs may add additional properties, arrays or objects to response payloads. To ensure maximum compatibility with future versions, calling applications should be coded to ignore elements that they do not recognize.

For example, suppose an application contains an endpoint foo that returns a response that looks like the following code snippet:

```
{
  "accountName":"test",
  "type":"individual",
  "status":"active"
}
```

At a later time, the response is modified to add creationDate and lastModifiedDate.

```
{
  "accountName":"test",
  "type":"individual",
  "status":"active",
  "creationDate":"2015-01-01 01:01:01",
  "lastModifiedDate":"2005-02-24 02:02:02"
}
```

Code to parse the response that was written to look for the original accountName, type, and status properties, will continue to work with the new response. However, the applications just would not take advantage of the new information.

For details, see the following topics:

- Administrators API
- Health Check API
- IP Address Whitelist API
- Management API
- Password Policy API
- Hybrid Data Pipeline API Error Messages
Administrators API

The Administrators API gives administrators control over resources used to provision users, manage roles and permissions, and manage other Hybrid Data Pipeline features.

Administrator Permissions API

The Administrator Permissions API is used to return a complete list of permissions or details on a particular permission. A user must have either the Administrator (12) or MgmtAPI (11) to use this API.

Permissions may be granted to roles, users, and data sources. The permissions for a user account are the sum of the permissions granted to the role(s) associated with the account and the permissions granted explicitly on the account. Any permissions specified on a data source will override the permissions for the user that owns the data source. (See also User provisioning on page 107.)

You can perform the following operations with the Administrator Permissions API.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve a complete list of supported permissions</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/permissions</td>
</tr>
<tr>
<td>Retrieve details about a permission</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/permissions/{id}</td>
</tr>
</tbody>
</table>

Get permissions

Purpose
Retrieves a complete list of supported permissions.

URL
https://<myserver>:<port>/api/admin/permissions

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition
The response takes the following format.

```json
{
  "permissions": [ 
```
ValidValuesDescriptionProperty
SeePermissionsanddefaultrolesonpage 59.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the permission.</td>
<td>See Permissions and default roles on page 59.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the permission.</td>
<td>See Permissions and default roles on page 59.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the permission.</td>
<td>See Permissions and default roles on page 59.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```
{
  "permissions": [
    {
      "id": 1,
      "name": "CreateDataSource",
      "description": "May create new data sources."
    },
    {
      "id": 2,
      "name": "ViewDataSource",
      "description": "May view any data source they own (when given to a role or user) or view an individual data source they own (when given to a data source)."
    },
    {
      "id": 3,
      "name": "ModifyDataSource",
      "description": "May modify/update any data source they own (when given to a role or user) or modify/update an individual data source they own (when given to a data source)."
    }
  ]
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222207919,
    "message": {
      "lang": "en-US",
      "value": "Problem getting Roles at this time. Please try again at another time."
    }
  }
}
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have either the Administrator (12) or MgmtAPI (11) permission.

Get details on a permission

Purpose
Retrieves details on a permission.

URL
https://<myserver>:<port>/api/admin/permissions/{id}

Method
GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of a permission.</td>
<td>See Permissions and default roles on page 59.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format.

```json
{
    "id": permission_id,
    "name": "permission_name",
    "description": "permission_description"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the permission</td>
<td>See Permissions and default roles on page 59.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the permission</td>
<td>See <a href="#">Permissions and default roles</a> on page 59.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the permission</td>
<td>See <a href="#">Permissions and default roles</a> on page 59.</td>
</tr>
</tbody>
</table>

### Sample Server Success Response

Status code: 200
Successful response

```json
{
    "id": 1,
    "name": "CreateDataSource",
    "description": "May create new data sources"
}
```

### Sample Server Failure Response

```json
{
    "error":{
        "code":222208553,
        "message":{
            "lang":"en-US",
            "value":"There is no Permission with that id: 1234"
        }
    }
}
```

### Authentication

**Basic Authentication using Login ID and Password**

### Authorization

The user must have either the Administrator (12) or MgmtAPI (11) permission.

### Authentication API

Hybrid Data Pipeline supports internal and external authentication services. When using the default internal authentication service, the end user authenticates directly with Hybrid Data Pipeline by passing the username and password for his or her Hybrid Data Pipeline user account. Alternatively, one or more end users can be associated with a Hybrid Data Pipeline user account through an external authentication service. In this case, end users pass credentials managed by the external service. Any end user who authenticates via an external service is in effect a proxy for the associated Hybrid Data Pipeline account, and inherits the permissions and administrative access given to the user account.
Administrators use the Authentication API to register external authentication services. An external authentication service can be registered with multiple tenants in the system. However, the service must be registered separately for each tenant. Once a service is registered with a tenant, the tenant administrator can provision end users in the tenant to authenticate via the service. A user with the Administrator (12) permission can register an external authentication service on any tenant within the system. A user with the RegisterExternalAuthService (26) permission can register an external authentication service on any tenant for which he or she has administrative access.

For detailed instructions on setting up external authentication services, see Authentication on page 143.

The following table summarizes operations that can be carried out with the Authentication API.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve authentication types</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/auth/types</td>
</tr>
<tr>
<td>Retrieve information on an authentication type</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/auth/types/{id}</td>
</tr>
<tr>
<td>Retrieve authentication services</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/auth/services</td>
</tr>
<tr>
<td>Register an external authentication service</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/auth/services</td>
</tr>
<tr>
<td>Retrieve information on an authentication service</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/auth/services/{id}</td>
</tr>
<tr>
<td>Update an authentication service</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/auth/services/{id}</td>
</tr>
<tr>
<td>Remove an authentication service</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/auth/services/{id}</td>
</tr>
</tbody>
</table>

### Get authentication types

#### Purpose
Retrieves supported authentication types.

#### URL
https://<myserver>:<port>/api/admin/auth/types

#### Method
GET

#### URL Parameters

- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
**Response Definition**
The response takes the following format.

```json
{
    "authTypes": [ 
        {
            "id": authtype_id,
            "name": "authtype_name",
            "description": "authtype_description"
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the authentication type.</td>
<td>1</td>
</tr>
</tbody>
</table>
|              |                                       | 1 is the ID for the internal authentication type.
|              |                                       | 2 is the ID for a service that uses a Java plugin.
|              |                                       | 3 is the ID for a service that uses LDAP.        |
| "name"       | The name of the authentication type.  | A string that specifies the name of the authentication type. |
| "description"| The description of the authentication type. | A string that provides the description of the authentication type. |

**Sample Server Success Response**

Status code: 200
Successful response

```json
{
    "authTypes": [ 
        {
            "id": 1,
            "name": "Internal",
            "description": "Password stored in service. The default HDP authentication."
        },
        {
            "id": 2,
            "name": "Java Auth Plugin",
            "description": "An authentication service that implements a Java Authentication plugin interface."
        },
        {
            "id": 3,
            "name": "LDAP Auth Plugin",
            "description": "An authentication service that authenticates User with LDAP."
        }
    ]
}
```

**Sample Server Failure Response**

Status code: 403
Forbidden
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have either the Administrator (12) permission, or the RegisterExternalAuthService (26) permission and administrative access to the tenant.

Get information on an authentication type

Purpose
Retrieves information on an authentication type.

URL
https://<myserver>:<port>/api/admin/auth/types/{id}

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the authentication type.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 is the ID for the internal authentication type.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 is the ID for a service that uses a Java plugin.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 is the ID for a service that uses LDAP.</td>
</tr>
</tbody>
</table>

Response Definition
The response has the following format.

```json
{
   "id": authtype_id,
   "name": "authtype_name",
   "description": "authtype_description",
   "authDefinition": {
      "className": "javaplugin_classname_info",
      "attributes": {authdefinition_attributes}
   }
}
```
<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the authentication type.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 is the ID for the internal authentication type.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 is the ID for a service that uses a Java plugin.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 is the ID for a service that uses LDAP.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication type.</td>
<td>A string that specifies the name of the authentication type.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication type.</td>
<td>A string that provides the description of the authentication type.</td>
</tr>
<tr>
<td>&quot;authDefinition&quot;</td>
<td>Information that describes the authentication type.</td>
<td>The value of authDefinition varies depending on which type is queried. A value of null is provided for the internal authentication service. See the example responses below.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

The response payload varies depending on which authentication type is queried.

**Internal authentication type**

Status code: 200  
Successful response

```
{
  "id": 1,
  "name": "Internal",
  "description": "Password stored in service. The default HDP authentication.",
  "authDefinition": null
}
```

**Java plugin authentication type**

Status code: 200  
Successful response

```
{
  "id": 2,
  "name": "Java Auth Plugin",
  "description": "An authentication service that implements a Java Authentication plugin interface.",
  "authDefinition": {
    "className": "Specify a concrete class name that implements Java Authentication Plugin Interface. Eg. com.sample.plugins.auth.JavaPluginAuthSample",
    "attributes": "This is optional. Attributes can take any valid JSON Object."
  }
}
```
LDAP authentication type

Status code: 200
Successful response

```
{
   "id": 3,
   "name": "LDAP Auth Plugin",
   "description": "An authentication service that authenticates User with LDAP.",
   "authDefinition": {
      "attributes": {
         "targetUrl": "<ldap server url>",
         "securityAuthentication": "<auth mechanism none,simple,sasl_mech>",
         "securityPrincipal": "<dn with loginname token>",
         "otherAttributes": "<This is Optional. JSON Object with key and value pairs which needs to be passed in environment properties while creating InitialDirContext obj>"
      }
   }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have either the Administrator (12) permission, or the RegisterExternalAuthService (26) permission and administrative access to the tenant.

Get authentication services

Purpose

Retrieves authentication services.

URL

https://<myserver>:<port>/api/admin/auth/services

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition

The response takes the following format.

```
{
   "authServices": [
      {
         "id": authservice_id,
```
"name": "authservice_id",
"tenantId": tenant_id,
"description": "authservice_description",
"tenantName": tenant_name
},
...
]}

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the authentication service.</td>
<td>The automatically generated external authentication service ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication service.</td>
<td>A string that provides a name for the authentication service.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication service.</td>
<td>A string that provides a description for the authentication service.</td>
</tr>
<tr>
<td>&quot;tenantName&quot;</td>
<td>The name of the tenant.</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
<tr>
<td></td>
<td>Only supplied when the URL is appended with the details query parameter set to true (?details=true).</td>
<td></td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```
{
   "authServices": [
   {
      "id": 1,
      "name": "Internal",
      "tenantId": 1,
      "description": "The default internal authentication service.",
      "tenantName": "System"
   },
   {
      "id": 21,
      "name": "LDAP",
      "tenantId": 43,
      "description": "LDAP Auth plugin",
      "tenantName": "OrgL"
   },
   {
      "id": 164,
      "name": "jauthplugi",
      "tenantId": 103,
      "description": "Java authentication plugin",
      "tenantName": "OrgR"
   }
   ]
}
```
Sample Server Failure Response

```
{
    "error": {
        "code": 222208103,
        "message": {
            "lang": "en-US",
            "value": "You lack the permissions to access this url."
        }
    }
}
```

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have either the Administrator (12) permission, or the RegisterExternalAuthService (26) permission and administrative access to the tenant.

Register external authentication service

Purpose
Registers an external authentication service. An external authentication service can be created using a Java plugin or LDAP.

URL

https://<myserver>:<port>/api/admin/auth/services

Method
POST

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Definition

The request payload definition varies depending on whether the service is a Java plugin service or an LDAP service.

Request definition for Java plugin service

```
{
    "name": "authservice_name",
    "tenantId": "tenant_id",
    "description": "authservice_description",
    "authDefinition": {
        "className": "java_plugin_classname",
        "attributes": {
            "attribute_name": "attribute_value",
            "attribute_name": "attribute_value",
        }
    }
}
```
... },
    "authTypeId": authtype_id
}

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication service.</td>
<td>Required</td>
<td>A string that provides a name for the authentication service.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant.</td>
<td>Optional</td>
<td>A valid tenant ID. If the tenant ID is not specified, the authentication service will belong to the tenant of the administrator executing the operation.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication service.</td>
<td>Optional</td>
<td>A string that provides a description for the authentication service.</td>
</tr>
<tr>
<td>&quot;authDefinition&quot;</td>
<td>An object that defines the authentication service.</td>
<td>Required</td>
<td>The authDefinition property must include the className property for a Java plugin service. The attributes property can provide useful information, such as an authentication server name, to be consumed by the authentication service. See authDefinition Object on page 1053 for details.</td>
</tr>
<tr>
<td>&quot;authTypeId&quot;</td>
<td>The ID of the authentication type.</td>
<td>Required</td>
<td>2 must be specified for a Java plugin service.</td>
</tr>
</tbody>
</table>

Request definition for LDAP service

```json
{
    "name": "authservice_name",
    "tenantId": tenant_id,
    "description": "authservice_description",
    "authDefinition": {
        "attributes": {
            "targetUrl": "LDAP_URL",
            "securityAuthentication": "LDAP_auth_mechanism",
            "securityPrincipal": "LDAP_principal",
            "securityCredentials": "LDAP_credentials"
        }
    },
    "authTypeId": authtype_id
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication service.</td>
<td>Required</td>
<td>A string that provides a name for the authentication service.</td>
</tr>
</tbody>
</table>
### Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant.</td>
<td>Optional</td>
<td>A valid tenant ID. If the tenant ID is not specified, the authentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>service will belong to the tenant of the administrator executing the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>operation.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication</td>
<td>Optional</td>
<td>A string that provides a description for the authentication service.</td>
</tr>
<tr>
<td>&quot;authDefinition&quot;</td>
<td>An object that defines the</td>
<td>Required</td>
<td>For an LDAP service, the following attributes must be specified via the</td>
</tr>
<tr>
<td></td>
<td>authentication service.</td>
<td></td>
<td>attributes object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- targetUrl</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- securityAuthentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- securityPrincipal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- securityCredentials (optional)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>See authDefinition Object on page 1053 for details.</td>
</tr>
<tr>
<td>&quot;authTypeId&quot;</td>
<td>The ID of the authentication type.</td>
<td>Required</td>
<td>3 must be specified for an LDAP service.</td>
</tr>
</tbody>
</table>

### Sample Request Payload

**Java plugin example request**

```json
{
  "name": "jplugauth",
  "tenantId": 1,
  "description": "Java external auth plugin",
  "authDefinition": {
    "className": "com.test.hdp.plugins.auth.HDPUserAuthentication",
    "attributes": {
      "Server": "test-authentication",
      "BackupServer": "test-authentication-backup"
    }
  },
  "authTypeId": 2
}
```

**LDAP example request**

```json
{
  "name": "LDAP",
  "tenantId": 66,
  "description": "LDAP Auth plugin",
  "authDefinition": {
    "attributes": {
      "targetUrl": "LDAP://123.45.67.899:389",
      "securityAuthentication": "simple",
      "securityPrincipal": "CN=%LOGINNAME%,OU=TestRuns,DC=testdomain,DC=local"
    }
  },
  "authTypeId": 3
}
```
Sample Response Payload
Java plugin example response

Status code: 201
Successful response

```
{
  "id": 43,
  "name": "jplugauth",
  "tenantId": 1,
  "description": "Java external auth plugin",
  "authDefinition": {
    "className": "com.test.hdp.plugins.auth.HDPUserAuthentication",
    "attributes": {
      "Server": "test-authentication",
      "BackupServer": "test-authentication-backup"
    }
  },
  "lastModifiedTime": "2018-02-15T11:09:35.107Z",
  "authTypeId": 2,
  "tenantName": "OrgM"
}
```

LDAP example response

Status code: 201
Successful response

```
{
  "id": 21,
  "name": "LDAP",
  "tenantId": 66,
  "description": "LDAP Auth plugin",
  "authDefinition": {
    "attributes": {
      "targetUrl": "LDAP://123.45.67.899:389",
      "securityAuthentication": "simple",
      "securityPrincipal": "CN=%LOGINNAME%,OU=TestRuns,DC=testdomain,DC=local"
    }
  },
  "lastModifiedTime": "2018-02-14T11:34:13.009Z",
  "authTypeId": 3,
  "tenantName": "OrgT"
}
```

Sample Server Failure Response

Status code: 400
Bad request, payload issues.

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have either the Administrator (12) permission, or the RegisterExternalAuthService (26) permission and administrative access to the tenant.
authDefinition Object

Purpose
Describes an external authentication service. Properties depend on whether the object describes a Java plugin service or an LDAP service.

Java plugin service
The authDefinition object for a Java plugin service consists of the className and attributes properties.

```
{
    "className": "java_plugin_classname",
    "attributes": {
        "attribute_name": "attribute_value",
        "attribute_name": "attribute_value",
        ...
    }
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;className&quot;</td>
<td>The class name that implements the Java authentication plugin interface.</td>
<td>The name of the class that the Java plugin developer created to implement the Java authentication plugin interface.</td>
</tr>
<tr>
<td>&quot;attributes&quot;</td>
<td>A JSON object comprised of named attribute values that are passed to the init method of the Java plugin. These attributes can provide useful values for initialization, such as an authentication server name, and can be used to configure the plugin for use by multiple authentication servers.</td>
<td>A valid JSON object</td>
</tr>
</tbody>
</table>

LDAP service
The authDefinition object for an LDAP service must include an attributes object consisting of the targetUrl, securityAuthentication, securityPrincipal, and securityCredentials attributes.

```
{
    "attributes": {
        "targetUrl": "LDAP_URL",
        "securityAuthentication": "LDAP_auth_mechanism",
        "securityPrincipal": "LDAP_principal",
        "securityCredentials": "LDAP_credentials"
    }
}
```

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;targetUrl&quot;</td>
<td>The URL used to access the LDAP server.</td>
<td>A string that specifies the URL for the LDAP server.</td>
</tr>
<tr>
<td>Attributes</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;securityAuthentication&quot;</td>
<td>The authentication mechanism required by the LDAP server.</td>
<td>none</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If none, an authentication mechanism is not used to authenticate against the LDAP server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If simple, a clear text password is used to authenticate against the LDAP server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If sasl_mech, the specified SASL authentication mechanism is used to authenticate against the LDAP server. For details, refer to Authentication Mechanisms in The Java Tutorials.</td>
</tr>
<tr>
<td>&quot;securityPrincipal&quot;</td>
<td>The principal used to authenticate against the LDAP server.</td>
<td>The principal information required will differ based on the authentication mechanism specified per the securityAuthentication attribute.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If none, this property is ignored.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If simple, the fully qualified domain name.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If sasl_mech, the SASL authorization identity. The authorization identity is the identity of the entity for which access control checks should be made if the authentication succeeds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> The username token %LOGINNAME% is supported to permit the replacement of the actual username. For example, CN=%LOGINNAME%,OU=TestRuns,DC=testdomain,DC=local.</td>
</tr>
<tr>
<td>&quot;securityCredentials&quot;</td>
<td>The credentials required to authenticate against the LDAP server.</td>
<td>The credential information required will differ based on the authentication mechanism specified per the securityAuthentication attribute.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If none, this property is ignored.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If simple, the password must be specified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If sasl_mech, the authorization credential key or password must be specified.</td>
</tr>
</tbody>
</table>

**Get information on authentication service**

**Purpose**

Retrieve information on an authentication service.
**URL**

https://<myserver>:<port>/api/admin/auth/services/{id}

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the authentication service.</td>
<td>The automatically generated external authentication service ID.</td>
</tr>
</tbody>
</table>

**Response Payload Definition**

The response payload definition varies depending on whether the service is a Java plugin service or an LDAP service.

**Response definition for Java plugin service**

```
{
    "name": "authservice_name",
    "tenantId": tenant_id,
    "description": "authservice_description",
    "authDefinition": {
        "className": "java_plugin_classname",
        "attributes": {
            "attribute_name": "attribute_value",
            "attribute_name": "attribute_value",
            ...
        },
        "lastModifiedTime": "timestamp",
        "authTypeId": authtype_id,
        "tenantName": tenant_name
    }
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication service.</td>
<td>A string that provides a name for the authentication service.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication service.</td>
<td>A string that provides a description for the authentication service.</td>
</tr>
</tbody>
</table>
The authDefinition property must include the className property for a Java plugin service. The attributes property can provide useful information, such as an authentication server name, to be consumed by the authentication service. See authDefinition Object on page 1053 for details.

A string that specifies the name of the tenant.

Valid Values

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;authDefinition&quot;</td>
<td>An object that defines the authentication service.</td>
<td>The authDefinition property must include the className property for a Java plugin service. The attributes property can provide useful information, such as an authentication server name, to be consumed by the authentication service. See authDefinition Object on page 1053 for details.</td>
</tr>
<tr>
<td>&quot;lastModifiedTime&quot;</td>
<td>The date and time the service was last modified.</td>
<td>A complete datetime with timezone string.</td>
</tr>
<tr>
<td>&quot;authTypeId&quot;</td>
<td>The ID of the authentication type.</td>
<td>2 must be specified for a Java plugin service.</td>
</tr>
<tr>
<td>&quot;tenantName&quot;</td>
<td>The name of the tenant.</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
</tbody>
</table>

Response definition for LDAP service

```
{
    "id": authservic_id,
    "name": authservIce_name",
    "tenantId": tenant_id,
    "description": "authservice_description",
    "authDefinition": {
        "attributes": {
            "targetUrl": "LDAP_URL",
            "securityAuthentication": "LDAP_auth_mechanism",
            "securityPrincipal": "LDAP_principal",
            "securityCredentials": "LDAP_credentials"
        }
    },
    "lastModifiedTime": "timestamp",
    "authTypeId": authtype_id,
    "tenantName": tenant_name
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication service.</td>
<td>A string that provides a name for the authentication service.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication service.</td>
<td>A string that provides a description for the authentication service.</td>
</tr>
</tbody>
</table>
Valid Values Description Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
</table>
| "authDefinition" | An object that defines the authentication service. | For an LDAP service, the following attributes must be specified via the attributes object.  
  • targetUrl  
  • securityAuthentication  
  • securityPrincipal  
  • securityCredentials (optional)  
  See authDefinition Object on page 1053 for details. |
| "lastModifiedTime" | The date and time the service was last modified. | A complete datetime with timezone string. |
| "authTypeId" | The ID of the authentication type. | 3 must be specified for an LDAP service. |
| "tenantName" | The name of the tenant. | A string that specifies the name of the tenant. |

Sample Response Payload

Java plugin example response

Status code: 200
Successful response

```
{
  "id": 43,
  "name": "jplugauth",
  "tenantId": 1,
  "description": "Java external auth plugin",
  "authDefinition": {
    "className": "com.test.hdp.plugins.auth.HDPUserAuthentication",
    "attributes": {
      "Server": "test-authentication",
      "BackupServer": "test-authentication-backup"
    }
  },
  "lastModifiedTime": "2018-02-15T11:09:35.107Z",
  "authTypeId": 2,
  "tenantName": "OrgM"
}
```

LDAP example response

Status code: 200
Successful response

```
{
  "id": 21,
  "name": "LDAP",
  "tenantId": 66,
  "description": "LDAP Auth plugin",
  "authDefinition": {
    "attributes": {
      "targetUrl": "LDAP://123.45.67.899:389",
      "securityAuthentication": "simple",
    }
  }
}
```
Sample Server Failure Response

Status code: 404
Supplied Services ID not found.

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have either the Administrator (12) permission, or the RegisterExternalAuthService (26) permission and administrative access to the tenant.

Update an authentication service

Purpose
Updates an authentication service. The internal authentication service cannot be modified.

URL
https://<myserver>:<port>/api/admin/auth/services/{id}

Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the authentication service.</td>
<td>The automatically generated external authentication service ID.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request payload definition varies depending on whether the service is a Java plugin service or an LDAP service.
Request definition for Java plugin service

```json
{
    "name": "authservice_name",
    "tenantId": tenant_id,
    "description": "authservice_description",
    "authDefinition": {
        "className": "java_plugin_classname",
        "attributes": {
            "attribute_name": "attribute_value",
            "attribute_name": "attribute_value",
            ...
        },
        "authTypeId": authtype_id
    }
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication service.</td>
<td>Required</td>
<td>A string that provides a name for the authentication service.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant.</td>
<td>Optional</td>
<td>A valid tenant ID. If the tenant ID is not specified, the authentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>service will belong to the tenant of the administrator executing the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>operation.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication</td>
<td>Optional</td>
<td>A string that provides a description for the authentication service.</td>
</tr>
<tr>
<td></td>
<td>service.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;authDefinition&quot;</td>
<td>An object that defines the authentication service.</td>
<td>Required</td>
<td>For an LDAP service, the following attributes must be specified via the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>attributes object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• targetUrl</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• securityAuthentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• securityPrincipal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• securityCredentials (optional)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>See <a href="#authdefinitionobject">authDefinition Object</a> on page 1053 for details.</td>
</tr>
<tr>
<td>&quot;authTypeId&quot;</td>
<td>The ID of the authentication type.</td>
<td>Required</td>
<td>3 must be specified for an LDAP service.</td>
</tr>
</tbody>
</table>

Request definition for LDAP service

```json
{
    "name": "authservice_name",
    "description": "authservice_description",
    "authDefinition": {
        "attributes": {
            "targetUrl": LDAP_URL,
            "securityAuthentication": "LDAP_auth_mechanism",
            "securityPrincipal": "LDAP_principal",
            "securityCredentials": "LDAP_credentials"
        }
    }
}
```
"authTypeId": authtype_id

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the authentication service.</td>
<td>Required</td>
<td>A string that provides a name for the authentication service.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the authentication service.</td>
<td>Optional</td>
<td>A string that provides a description for the authentication service.</td>
</tr>
<tr>
<td>&quot;authDefinition&quot;</td>
<td>An object that defines the authentication service.</td>
<td>Required</td>
<td>For an LDAP service, the following attributes must be specified via the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>attributes object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• targetUrl</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• securityAuthentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• securityPrincipal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• securityCredentials (optional)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>See authDefinition Object on page 1053 for details.</td>
</tr>
<tr>
<td>&quot;authTypeId&quot;</td>
<td>The ID of the authentication type.</td>
<td>Required</td>
<td>3 must be specified for an LDAP service.</td>
</tr>
</tbody>
</table>

Sample Request Payload

Java plugin example request

```json
{
    "name": "jplugauth",
    "tenantId": 1,
    "description": "Java external auth plugin",
    "authDefinition": {
        "className": "com.prod.hdp.plugins.auth.HDPUserAuthentication",
        "attributes": {
            "Server": "prod-authentication",
            "BackupServer": "prod-authentication-backup"
        }
    },
    "authTypeId": 2
}
```

LDAP example request

```json
{
    "name": "LDAP",
    "tenantId": 66,
    "description": "LDAP Auth plugin",
    "authDefinition": {
        "attributes": {
            "targetUrl": "LDAP://987.65.43.211:389",
            "securityAuthentication": "simple",
            "securityPrincipal": "CN=%LOGINNAME%,OU=ProdRuns,DC=proddomain,DC=local"
        }
    },
    "authTypeId": 3
}
```
Sample Response Payload
Java plugin example response

Status code: 200
Successful response

```
{
   "id": 43,
   "name": "jplugauth",
   "tenantId": 1,
   "description": "Java external auth plugin",
   "authDefinition": {
      "className": "com.prod.hdp.plugins.auth.HDPUserAuthentication",
      "attributes": {
         "Server": "prod-authentication",
         "BackupServer": "prod-authentication-backup"
      }
   },
   "lastModifiedTime": "2018-02-15T11:09:35.107Z",
   "authTypeId": 2,
   "tenantName": "OrgM"
}
```

LDAP example response

Status code: 200
Successful response

```
{
   "id": 21,
   "name": "LDAP",
   "tenantId": 66,
   "description": "LDAP Auth plugin",
   "authDefinition": {
      "attributes": {
         "targetUrl": "LDAP://987.65.43.211:389",
         "securityAuthentication": "simple",
         "securityPrincipal": "CN=%LOGINNAME%,OU=ProdRuns,DC=proddomain,DC=local"
      }
   },
   "lastModifiedTime": "2018-02-14T11:34:13.009Z",
   "authTypeId": 3,
   "tenantName": "OrgT"
}
```

Sample Server Failure Response

Status code: 404
Supplied Services ID not found.

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have either the Administrator (12) permission, or the RegisterExternalAuthService (26) permission and administrative access to the tenant.
Delete an authentication service

Purpose
Removes an authentication service. The internal authentication service cannot be deleted.

URL
https://<myserver>:<port>/api/admin/auth/services/{id}

Method
DELETE

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the authentication service.</td>
<td>The automatically generated external authentication service ID.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Request fulfilled

Sample Server Failure Response

```json
{
   "error": {
      "code": 222208085,
      "message": {
         "lang": "en-US",
         "value": "There is no Auth Service with the Id: 77."
      }
   }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have either the Administrator (12) permission, or the RegisterExternalAuthService (26) permission and administrative access to the tenant.
CORS Whitelist API

Hybrid Data Pipeline supports cross-origin resource sharing (CORS) filters that allow the sharing of web resources across domains. Configuring CORS behavior is a two part process. An administrator must enable CORS behavior via the Limits API and create a whitelist of trusted origins. The CORS Whitelist API is used to create and manage a CORS whitelist. (See also Configuring CORS behavior on page 172.)

To create and manage a whitelist, the administrator must have either the Administrator (12) permission or the CORS whitelist (23) permission and administrative access on the default system tenant.

The following table summarizes the operations that are supported with the CORS Whitelist API.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve the CORS whitelist</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/cors/whitelist</td>
</tr>
<tr>
<td>Create a CORS whitelist or add trusted origins to a CORS whitelist</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/cors/whitelist</td>
</tr>
<tr>
<td>Retrieve information on a trusted origin</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/cors/whitelist/{id}</td>
</tr>
<tr>
<td>Update information on a trusted origin</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/cors/whitelist/{id}</td>
</tr>
<tr>
<td>Delete a trusted origin</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/cors/whitelist/{id}</td>
</tr>
</tbody>
</table>

Get CORS whitelist

Purpose

Retrieves the CORS whitelist. The whitelist is an array of JSON objects. Each object, or entry, provides details for each trusted origin.

URL

https://<myserver>:<port>/api/admin/security/cors/whitelist

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
Response Payload Definition

The response payload takes the following format.

```
{
    "lastModifiedTime": "timestamp",
    "whitelist": [
        {
            "id": "trusted_origin_id",
            "domain": "trusted_origin_domain",
            "description": "domain_description",
            "lastModifiedBy": "username",
            "lastModifiedTime": "timestamp"
        },
        ...
    ]
}
```

The `lastModifiedTime` property indicates the last time the whitelist was modified.

The `whitelist` property is an array of JSON objects. Each object, or entry, provides details (described in the following table) for each trusted origin.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the trusted origin.</td>
<td>An unique ID that is generated when a trusted origin is added to the CORS whitelist.</td>
</tr>
<tr>
<td>&quot;domain&quot;</td>
<td>The domain of the trusted origin.</td>
<td>A valid domain for the trusted origin. For example, <a href="https://abc.com">https://abc.com</a>. The wild card * can be used at the beginning of a domain. For example, <em>.progress.com is a valid entry, and will whitelist any origin that ends with progress.com. The wild card is not supported at any other location within a domain. For example, progress.abc.</em>.com is not supported for origin validation.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the trusted origin.</td>
<td>A user provided description of the trusted origin.</td>
</tr>
<tr>
<td>&quot;lastModifiedBy&quot;</td>
<td>The name of the administrator who last modified the entry of the trusted origin.</td>
<td>The Hybrid Data Pipeline username of the administrator.</td>
</tr>
<tr>
<td>&quot;lastModifiedTime&quot;</td>
<td>The last time the entry of the trusted origin was modified.</td>
<td>A timestamp.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```
Status code: 200
Successful response

{
    "lastModifiedTime": "2017-08-13T18:15:09.352Z",
    "whitelist":
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the CORS whitelist (23) permission.

Create whitelist or add trusted origins to whitelist

Purpose
Creates a CORS whitelist or adds trusted origins to a CORS whitelist. The whitelist is an array of JSON objects. Each object, or entry, provides details for each trusted origin.

URL
https://<myserver>:<port>/api/admin/security/cors/whitelist

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Definition
The request payload takes the following format.

```json
{
    "whitelist": [
        {
            "domain": "trusted_origin_domain",
            "description": "domain_description"
        },
        ...
    ]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;domain&quot;</td>
<td>The domain of the trusted origin.</td>
<td>Required</td>
<td>A valid domain for the trusted origin. For example, <a href="https://abc.com">https://abc.com</a>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The wildcard * can be used at the beginning of a domain. For example, <em>_progress.com is a valid entry, and will whitelist any origin that ends with progress.com. The wildcard is not supported at any other location within a domain. For example, progress.abc.</em>_.com is not supported for origin validation.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the trusted origin.</td>
<td>Optional</td>
<td>A user provided description of the trusted origin.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```
{
   "whitelist": [
   {
      "domain": "http://*.abc.com",
      "description": "The ABC group domain"
   },
   {
      "domain": "http://bar.test.com",
      "description": "The bar trusted origin"
   }
   ]
}
```

**Sample Server Success Response**

Status code: 201
Successful response

```
{
   "whitelist": [
   {
      "domain": "http://*.abc.com",
      "description": "The ABC group domain"
   },
   {
      "domain": "http://bar.test.com",
      "description": "The bar trusted origin"
   }
   ]
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) or the CORSwhitelist (23) permission.
Get information on a trusted origin

Purpose
Retrieve information on a trusted origin for a CORS whitelist.

URL
https://<myserver>:<port>/api/admin/security/cors/whitelist/{id}

Method
GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following {id} parameter is required in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the trusted origin.</td>
<td>An unique ID that is generated when a trusted origin is added to the CORS whitelist.</td>
</tr>
</tbody>
</table>

Response Payload Definition

The response payload takes the following format.

```json
{
  "id": trusted_origin_id,
  "domain": "trusted_origin_domain",
  "description": "domain_description",
  "lastModifiedBy": "username",
  "lastModifiedTime": "timestamp"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the trusted origin.</td>
<td>An unique ID that is generated when a trusted origin is added to the CORS whitelist.</td>
</tr>
</tbody>
</table>
## Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;domain&quot;</td>
<td>The domain of the trusted origin.</td>
<td>A valid domain for the trusted origin. For example, <a href="https://abc.com">https://abc.com</a>. The wild card * can be used at the beginning of a domain. For example, <em>.progress.com is a valid entry, and will whitelist any origin that ends with progress.com. The wild card is not supported at any other location within a domain. For example, progress.abc.</em>.com is not supported for origin validation.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the trusted origin.</td>
<td>A user provided description of the trusted origin.</td>
</tr>
<tr>
<td>&quot;lastModifiedBy&quot;</td>
<td>The name of the administrator who last modified the entry of the trusted origin.</td>
<td>The Hybrid Data Pipeline username of the administrator.</td>
</tr>
<tr>
<td>&quot;lastModifiedTime&quot;</td>
<td>The last time the entry of the trusted origin was modified.</td>
<td>A timestamp.</td>
</tr>
</tbody>
</table>

### Sample Server Success Response

Status code: 200
Successful response

```json
{
   "id": 1,
   "domain": "http://*.abc.com",
   "description": "The ABC group domain",
   "lastModifiedBy": "Admin1",
   "lastModifiedTime": "2017-08-13T18:15:09.352Z"
}
```

### Authentication

Basic Authentication using Login ID and Password

### Authorization

The user must have the Administrator (12) or the CORS whitelist (23) permission.

### Update information on a trusted origin

#### Purpose

Updates the information on a trusted origin.

#### URL

https://<myserver>:<port>/api/admin/security/cors/whitelist/{id}
Method

PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following {id} parameter is required in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the trusted origin.</td>
<td>An unique ID that is generated when a trusted origin is added to the CORS whitelist.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request payload takes the following format.

```json
{
    "domain": "trusted_origin_domain",
    "description": "domain_description"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;domain&quot;</td>
<td>The domain of the trusted origin.</td>
<td>Required</td>
<td>A valid domain for the trusted origin. For example, <a href="https://abc.com">https://abc.com</a>. The wildcard * can be used at the beginning of a domain. For example, <em>.progress.com is a valid entry, and will whitelist any origin that ends with progress.com. The wildcard is not supported at any other location within a domain. For example, progress.abc.</em>.com is not supported for origin validation.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the trusted origin.</td>
<td>Optional</td>
<td>A user provided description of the trusted origin.</td>
</tr>
</tbody>
</table>

Sample Request payload

```json
{
    "domain": "http://*.test.com",
    "description": "The ABC group domain"
}
```
Sample Server Success Response

Status code: 201
Successful response

{  
  "domain": "http://*.test.com",
  "description": "The test domain"
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the CORS whitelist (23) permission.

Delete a trusted origin on the CORS whitelist

Purpose
Delete a trusted origin on the CORS whitelist. (The entry remains on the whitelist but is marked as deleted.)

URL
https://<myserver>:<port>/api/admin/security/cors/whitelist/{id}

Method
DELETE

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following {id} parameter is required in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the trusted origin.</td>
<td>An unique ID that is generated when a trusted origin is added to the CORS whitelist.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

Authentication
Basic Authentication using Login ID and Password
Authorization

The user must have the Administrator (12) or the CORS whitelist (23) permission.

Limits API

The Limits API can be used to manage a number of Hybrid Data Pipeline features. For example, the Limits API can be used to restrict the number of rows in a query, or to implement an account lockout policy, or to enable CORS behavior. Each limit has a default value that governs some aspect of behavior related to its corresponding feature. Limits can be set at four levels: system, tenant, user, and data source. The following hierarchy applies to these levels.

1. Data source
2. User
3. Tenant
4. System

Limits set on a data source override limits set at the other levels; limits set on a user account override those set on a tenant or set at the system level; limits set on a tenant override those set at the system level; and limits set at the system level override default behavior. Default and system limits apply to behavior across Hybrid Data Pipeline, while limits on data sources, users, and tenants apply to the resources they handle. Most limits can only be configured at the system level. However, some limits, such as MaxFetchRows and ODataMaxConcurrentQueries, can be configured at any level.

The following tables provide summary information on the Limits API.

- The Supported limits table lists all configurable limits, their IDs, what levels they may be applied to, and their descriptions.
- The Limits API operations table lists supported operations with links to operation-specific topics for details.

Table 212: Supported limits

<table>
<thead>
<tr>
<th>Limits</th>
<th>ID</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxFetchRows</td>
<td>1</td>
<td>All levels</td>
<td>Maximum number of rows allowed to be fetched for a single query.</td>
</tr>
<tr>
<td>PasswordLockoutInterval</td>
<td>2</td>
<td>System level</td>
<td>The duration, in seconds, for counting the number of consecutive failed authentication attempts.</td>
</tr>
<tr>
<td>PasswordLockoutLimit</td>
<td>3</td>
<td>System level</td>
<td>The number of consecutive failed authentication attempts that are allowed before locking the user account.</td>
</tr>
<tr>
<td>PasswordLockoutPeriod</td>
<td>4</td>
<td>System level</td>
<td>The duration, in seconds, for which a user account will not be allowed to authenticate to the system when the PasswordLockoutLimit is reached.</td>
</tr>
<tr>
<td>Limits</td>
<td>ID</td>
<td>Usage</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CORSBehavior</td>
<td>5</td>
<td>System level</td>
<td>Configuration parameter for CORS behavior. Setting the value to 0 disables the CORS filter. Setting the value to 1 enables the CORS filter. Setting the value to 2 enables the CORS filter with the whitelist option.</td>
</tr>
<tr>
<td>ODataMaxConcurrentQueries</td>
<td>6</td>
<td>All levels</td>
<td>Maximum number of concurrent active OData queries per data source.</td>
</tr>
<tr>
<td>LogRetentionDays</td>
<td>7</td>
<td>System level</td>
<td>Number of days log files should be retained.</td>
</tr>
<tr>
<td>OAuthAccessTokenDuration</td>
<td>8</td>
<td>System level</td>
<td>The duration, in minutes, for which a Access token is valid.</td>
</tr>
<tr>
<td>MonitorRetentionDays</td>
<td>9</td>
<td>System level</td>
<td>Number of days monitor details should be retained</td>
</tr>
<tr>
<td>UserMeterRetentionDays</td>
<td>10</td>
<td>System level</td>
<td>Number of days user meter details should be retained</td>
</tr>
<tr>
<td>UserMeterWriteInterval</td>
<td>11</td>
<td>System level</td>
<td>The number of seconds the system waits before scanning sessions for current metrics. A lower setting will result in more rows written to the meter table</td>
</tr>
<tr>
<td>UserMeterMaxAge</td>
<td>12</td>
<td>System level</td>
<td>The number seconds the system waits before writing out meter records. A lower setting will result in the rows written to meter table to occur more frequently</td>
</tr>
<tr>
<td>OAuthAccessTokenCacheSize</td>
<td>13</td>
<td>System level</td>
<td>Number of oauth access tokens to be cached in memory for OAuth Authentication. By default up to 2000 tokens will be cached in memory.</td>
</tr>
<tr>
<td>TransactionTimeout</td>
<td>14</td>
<td>All levels</td>
<td>The number of seconds the system allows a transaction to be idle before rolling it back.</td>
</tr>
<tr>
<td>XdbcMaxResponse</td>
<td>15</td>
<td>All levels</td>
<td>Approximate maximum size of JDBC/ODBC HTTP result data in KB.</td>
</tr>
</tbody>
</table>

Table 213: Limits API operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve configurable limits</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits</td>
</tr>
<tr>
<td>Operation</td>
<td>Request</td>
<td>URL</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Retrieve limits that have been set at the system level</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/system</td>
</tr>
<tr>
<td>Retrieve a limit set at the system level</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/system/{limitId}</td>
</tr>
<tr>
<td>Create a limit at the system level</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/system/{limitId}</td>
</tr>
<tr>
<td>Update a limit set at the system level</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/system/{limitId}</td>
</tr>
<tr>
<td>Remove a limit set at the system level</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/system/{limitId}</td>
</tr>
<tr>
<td>Retrieve limits that have been set for tenants</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/tenants</td>
</tr>
<tr>
<td>Retrieve a limit set on a tenant</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/tenants/{tenantId}/{limitId}</td>
</tr>
<tr>
<td>Create a limit on a tenant</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/tenants/{tenantId}/{limitId}</td>
</tr>
<tr>
<td>Update a limit set on a tenant</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/tenants/{tenantId}/{limitId}</td>
</tr>
<tr>
<td>Remove a limit set on a tenant</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/tenants/{tenantId}/{limitId}</td>
</tr>
<tr>
<td>Retrieve limits that have been set on user accounts</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/users</td>
</tr>
<tr>
<td>Retrieve a limit set on a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/users/{userId}/{limitId}</td>
</tr>
<tr>
<td>Create a limit on a user account</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/users/{userId}/{limitId}</td>
</tr>
<tr>
<td>Update a limit on a user account</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/users/{userId}/{limitId}</td>
</tr>
<tr>
<td>Delete a limit set on a user account</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/users/{userId}/{limitId}</td>
</tr>
<tr>
<td>Retrieve limits that have been set on data sources for a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/limits/users/{userId}/datasources</td>
</tr>
</tbody>
</table>

Administrators API
### Get limits

**Purpose**

Retrieves configurable limits.

**URL**

https://<myserver>:<port>/api/admin/limits

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Definition**

The response takes the following format.

```json
{
    "limits": [ 
        {
            "id": limit_id,
            "name": "limit_name",
            "description": "limit_description",
            "minValue": min_value,
            "maxValue": max_value,
            "defaultValue": default_value,
            "validForLimits": integer 
        }, ...
    ]
}
```
### Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the limit.</td>
<td>A valid limit ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the limit.</td>
<td>A valid limit name.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the limit.</td>
<td>The limit description.</td>
</tr>
<tr>
<td>&quot;minValue&quot;</td>
<td>The minimum possible value of the limit.</td>
<td>A valid minimum value.</td>
</tr>
<tr>
<td>&quot;maxValue&quot;</td>
<td>The maximum possible value of the limit.</td>
<td>A valid maximum value.</td>
</tr>
<tr>
<td>&quot;defaultValue&quot;</td>
<td>The default value of the limit.</td>
<td>The default value.</td>
</tr>
<tr>
<td>&quot;validForLimits&quot;</td>
<td>A numeric value that indicates at what level or levels the limit can be set.</td>
<td>1</td>
</tr>
</tbody>
</table>

1 indicates the limit can only be valid at the system level.
15 indicates the limit can be set at the system, tenant, user, data source level, or a combination of these.

---

**Sample Server Success Response**

Status code: 200
Successful response

```json
{
  "limits": [
    {
      "id": 1,
      "name": "MaxFetchRows",
      "description": "Maximum number of rows allowed to be fetched for a single query",
      "minValue": 1,
      "maxValue": 9000000000000000000,
      "defaultValue": 9000000000000000000,
      "validForLimits": 15
    },
    {
      "id": 2,
      "name": "PasswordLockoutInterval",
      "description": "The duration, in seconds, for counting the number of consecutive failed authentication attempts.",
      "minValue": 1,
      "maxValue": 1000000000,
      "defaultValue": 900,
      "validForLimits": 1
    },
    {
      "id": 3,
      "name": "PasswordLockoutLimit",
      "description": "The number of consecutive failed authentication attempts that are allowed before locking the user account.",
      "minValue": 0,
      "maxValue": 9000000000000000000,
      "defaultValue": 9000000000000000000,
      "validForLimits": 15
    }
  ]
}```
"maxValue": 1000000,
"defaultValue": 3,
"validForLimits": 1
},
{
  "id": 4,
  "name": "PasswordLockoutPeriod",
  "description": "The duration, in seconds, for which a user account will not be allowed to authenticate to the system when the PasswordLockoutLimit is reached."
},
{
  "id": 5,
  "name": "CORSBehavior",
  "description": "Configuration parameter for CORS behavior. Setting the value to 0 disables the CORS filter. Setting the value to 1 enables the CORS filter. Setting the value to 2 enables the CORS filter with the whitelist option."
},
{
  "id": 6,
  "name": "ODataMaxConcurrentQueries",
  "description": "Maximum number of concurrent active queries per data source"
},
{
  "id": 7,
  "name": "LogRetentionDays",
  "description": "Number of days log files should be retained"
},
{
  "id": 8,
  "name": "OAuthAccessTokenDuration",
  "description": "The duration, in minutes, for which an Access token is valid."
},
{
  "id": 9,
  "name": "MonitorRetentionDays",
  "description": "Number of days monitor details should be retained"
},
{
  "id": 10,
  "name": "UserMeterRetentionDays",
  "description": "Number of days user meter details should be
Sample Server Failure Response

{
  "error": {
    "code": 222207925,
    "message": {
      "lang": "en-US",
      "value": "Problem processing the limits at this time. Please try again at another time."}
  }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the Limits (27) permission.

Get limits set at the system level

Purpose
Retrieves limits that have been set at the system level.

URL
https://<myserver>:\<port>/api/admin/limits/system
Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition
The response takes the following format.

```
{
   "limits": [
     {
       "id": limit_id,
       "value": limit_value
     },
     ...
   ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the limit.</td>
<td>A valid limit ID.</td>
</tr>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the system limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```
{
   "limits": [
     {
       "id": 1,
       "value": 500
     },
     {
       "id": 2,
       "value": 1800
     },
     {
       "id": 3,
       "value": 5
     },
     {
       "id": 4,
       "value": 3600
     },
     {
       "id": 6,
       "value": 1000
     },
     {
       "id": 8,
       "value": 60
     }
   ]
}
```
Sample Server Failure Response

```json
{
    "error": {
        "code": 222207925,
        "message": {
            "lang": "en-US",
            "value": "Problem processing the limits at this time. Please try again at another time.."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) or the Limits (27) permission.

**Get a limit set at the system level**

**Purpose**

Retrieves the value of a limit set at the system level.

**URL**

https://<myserver>:<port>/api/admin/limits/system/{limitId}

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {limitId} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>A valid limit ID.</td>
</tr>
</tbody>
</table>
Response Definition

The response takes the following format.

```
{
    "value": limit_value
}
```

### Property | Description | Valid Values
--- | --- | ---
"value" | The value of the limit. | An integer that meets the requirements of the minimum and maximum values for the limit.

Sample Server Success Response

```
Status code: 200
Successful response

{
    "value": 400
}
```

Sample Server Failure Response

```
{
    "error": {
        "code": 222207925,
        "message": {
            "lang": "en-US",
            "value": "Problem processing the limits at this time. Please try again at another time."}
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) or the Limits (27) permission.

Create a limit at the system level

**Purpose**

Creates a limit at the system level.

**URL**

https://<myserver>:<port>/api/admin/limits/system/{limitId}
Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {limitId} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>A valid limit ID.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format.

```json
{
    "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```json
{
    "value": 400
}
```

Sample Server Success Response

- Status code: 201
- Successful response

```json
{
    "value": 400
}
```

Sample Server Failure Response

```json
{
    "error": {
        "code": 222207929,
        "message": {
            "lang": "en-US",
            "value": "Limit value not in range({0}, {1})."
        }
    }
}
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission.

Update a limit set at the system level

Purpose
Updates a limit set at the system level.

URL
https://<myserver>:<port>/api/admin/limits/system/{limitId}

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {limitId} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>A valid limit ID.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format.

```json
{
    "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>
Sample Request Payload

```json
{
   "value": 400
}
```

Sample Server Success Response

```
Status code: 200
Successful response

{
   "value": 400
}
```

Sample Server Failure Response

```
{
   "error": {
      "code": 222207929,
      "message": {
         "lang": "en-US",
         "value": "Limit value not in range({0}, {1})."
      }
   }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission.

Delete a limit at the system level

Purpose

Deletes a limit set at the system level.

URL

https://<myserver>:<port>/api/admin/limits/system/{limitId}

Method

DELETE

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The URL parameter \{limitId\} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>A valid limit ID.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

- Status code: 204
- Successful response

**Sample Server Failure Response**

```json
{
  "error": {
    "code": 222207930,
    "message": {
      "lang": "en-US",
      "value": "Limit does not exist for id: \{1\}. "
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission.

**Get limits set on tenants**

**Purpose**

Retrieves limits that have been set on tenants.

**URL**

https://<myserver>:<port>/api/admin/limits/tenants

**Method**

GET

**URL Parameters**

- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
Response Definition

The response takes the following format.

```json
{
    "tenantLimits": [ 
        { 
            "tenantId": tenant_id,
            "tenantName": tenant_name,
            "limits": [ 
                { 
                    "id": limit_id,
                    "value": limit_value
                },
                ...
            ],
            ...
        }
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;tenantName&quot;</td>
<td>The name of the tenant.</td>
<td>The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;limits&quot;</td>
<td>A list of limits that have been set on the tenant. Includes the &quot;id&quot; and &quot;value&quot; properties where the &quot;id&quot; is the ID of the limit, and &quot;value&quot; is the value of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a tenant.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```json
{
    "tenantLimits": [ 
        { 
            "tenantId": 1,
            "tenantName": "System",
            "limits": [ ]
        },
        { 
            "tenantId": 71,
            "tenantName": "OrgA",
            "limits": [ 
                { 
                    "id": 1,
                    "value": 1000
                },
                { 
                    "id": 6,
                    "value": 100
                }
            ]
        }
    ]
}
```
Sample Server Failure Response

```json
{
   "error": {
      "code": 222207925,
      "message": {
         "lang": "en-US",
         "value": "Problem processing the limits at this time. Please try again at another time."
      }
   }
}
```

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the Limits (27) permission. In addition, limits are only returned for tenants that the user has access to administer. Any user with the Administrator permission will see limits set on all tenants, while a user with the Limits permission needs administrative access on a given tenant to see that tenant's limits.

Get a limit set on a tenant

Purpose
Retrieves the value for a limit that has been set on a tenant.

URL

```
https://<myserver>:<port>/api/admin/limits/tenants/{tenantId}/{limitId}
```

Method
GET

URL Parameters

- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{tenantId}</code></td>
<td>The ID of the tenant.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td><code>{limitId}</code></td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a tenant.</td>
</tr>
</tbody>
</table>
Response Definition
The response takes the following format.

```json
{
   "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```
Status code: 200
Successful response

{
   "value": 400
}
```

Sample Server Failure Response

```
{
   "error": {
      "code": 222207916,
      "message": {
         "lang": "en-US",
         "value": "There is no User with that id: 123."
      }
   }
}
```

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the Limits (27) permission.

Create a limit on a tenant

Purpose
Creates a limit on a tenant.

URL
https://<myserver>:<port>/api/admin/limits/tenants/{tenantId}/{limitId}

Method
POST
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{tenantId}</td>
<td>The ID of the tenant.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request takes the following format.

```
{
    "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```
{
    "value": 400
}
```

Sample Server Success Response

```
Status code: 201
Successful response

{
    "value": 400
}
```

Sample Server Failure Response

```
{
    "error": {
        "code": 222207929,
        "message": {
            "lang": "en-US",
            "value": 400
        }
    }
```
"value": "Limit value not in range({0}, {1})."
}
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the Limits (27) permission and administrative access on the tenant for which the limit is being set.

Update a limit on a tenant

Purpose
Updates a limit set on a tenant.

URL
https://<myserver>:<port>/api/admin/limits/tenants/{tenantId}/{limitId}

Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{tenantId}</td>
<td>The ID of the tenant.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>
**Request Payload Definition**

The request takes the following format.

```
{
   "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```
{
   "value": 400
}
```

**Sample Server Success Response**

```
Status code: 200
Successful response

{
   "value": 400
}
```

**Sample Server Failure Response**

```
{
   "error": {
      "code": 222207929,
      "message": {
         "lang": "en-US",
         "value": "Limit value not in range({0}, {1})."
      }
   }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission, or the Limits (27) permission and administrative access on the tenant for which the limit is being set.

**Delete a limit on a tenant**

**Purpose**

Removes a limit that was set on a tenant.
URL
https://<myserver>:<port>/api/admin/limits/tenants/{tenantId}/{limitId}

Method
DELETE

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{tenantId}</td>
<td>The ID of the tenant.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

Sample Server Failure Response

```json
{
  "error": {
    "code": 222207028,
    "message": {
      "lang": "en-US",
      "value": "Missing 'userId' in payload."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the Limits (27) permission and administrative access on the tenant for which the limit is being set.
Get limits set on user accounts

Purpose
Retrieves limits that have been set on user accounts.

URL
https://<myserver>:<port>/api/admin/limits/users

Method
GET

URL Parameters
<br/> <myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition
The response takes the following format.

```json
{
    "userLimits": [
        {
            "userId": user_id,
            "userName": "user_name",
            "limits": [
                {
                    "id": limit_id,
                    "value": limit_value
                },
                ...
            ]
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userId&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;userName&quot;</td>
<td>The name of the user account.</td>
<td>The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;limits&quot;</td>
<td>A list of limits that have been set on the user account. Includes the &quot;id&quot; and &quot;value&quot; properties where the &quot;id&quot; is the ID of the limit, and &quot;value&quot; is the value of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

Status code: 200
Successful response

```json
{
  "userLimits": [
    {
      "userId": 203,
      "userName": "user1",
      "limits": [
        {
          "id": 1,
          "value": 1000
        },
        {
          "id": 6,
          "value": 100
        }
      ]
    },
    {
      "userId": 204,
      "userName": "user2",
      "limits": []
    },
    ...
  ]
}
```

Sample Server Failure Response

```json
{
  "error": {
    "code": 222207925,
    "message": {
      "lang": "en-US",
      "value": "Problem processing the limits at this time. Please try again at another time."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the Limits (27) permission.

Get a limit set on a user account

Purpose

Retrieves the value for a limit that has been set on a user account.

URL

https://<myserver>:<port>/api/admin/limits/users/{userId}/{limitId}
Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format.

```json
{
   "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```json
{
   "value": 400
}
```

Sample Server Failure Response

```json
{
   "error": {
      "code": 222207916,
      "message": {
         "lang": "en-US",
         "value": "There is no User with that id: 123."
      }
   }
}
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the Limits (27) permission.

Create a limit on a user account

Purpose
 Creates a limit on a user account.

URL
https://<myserver>:<port>/api/admin/limits/users/{userId}/{limitId}

Method
POST

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>
Request Payload Definition

The request takes the following format.

```json
{
    "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```json
{
    "value": 400
}
```

Sample Server Success Response

Status code: 201
Successful response

```json
{
    "value": 400
}
```

Sample Server Failure Response

```json
{
    "error": {
        "code": 222207929,
        "message": {
            "lang": "en-US",
            "value": "Limit value not in range({0}, {1})."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the Limits (27) permission.

Update a limit on a user account

Purpose

Updates a limit that was set on a user account.
**URL**

https://<myserver>:<port>/api/admin/limits/users/{userId}/{limitId}

**Method**

PUT

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>

**Request Payload Definition**

The request takes the following format.

```json
{
    "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```json
{
    "value": 400
}
```

**Sample Server Success Response**

```
Status code: 200
Successful response

{
    "value": 400
}
```
Sample Server Failure Response

{
  "error": {
    "code": 222207929,
    "message": {
      "lang": "en-US",
      "value": "Limit value not in range({0}, {1})."
    }
  }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the Limits (27) permission.

Delete a limit set on a user account

Purpose

Deletes a limit that was set on a user account.

URL

https://<myserver>:<port>/api/admin/limits/users/{userId}/{limitId}

Method

DELETE

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a user account.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

Sample Server Failure Response

```
{
  "error": {
    "code": 222207028,
    "message": {
      "lang": "en-US",
      "value": "Missing 'userId' in payload."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the Limits (27) permission.

Get limits set on data sources for a user account

Purpose

Retrieves limits that have been set on data sources which belong to a specified user account.

URL

https://<myserver>:<port>/api/admin/limits/users/{userId}/datasources

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account to which the data sources belong.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Response Definition

The response takes the following format.

```json
{
    "dataSourceLimits": [
        {
            "dataSourceId": datasource_id,
            "dataSourceName": "dataSource_name",
            "isGroup": true | false,
            "limits": [  
                {
                    "id": limit_id,
                    "value": limit_value
                },
                ...
            ]
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;dataSourceId&quot;</td>
<td>The ID of the data source.</td>
<td>The data source ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;dataSourceName&quot;</td>
<td>The name of the data source.</td>
<td>The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;isGroup&quot;</td>
<td>Indicates whether the data source is a group data source.</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>true if the data source is a group data source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>false if the data source is not a group data source.</td>
</tr>
<tr>
<td>&quot;limits&quot;</td>
<td>A list of limits that have been set on the data source. Includes the &quot;id&quot; and &quot;value&quot; properties where the &quot;id&quot; is the ID of the limit, and &quot;value&quot; is the value of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a data source.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

```json
{
    "dataSourceLimits": [
        {
            "dataSourceId": 1,
            "dataSourceName": "DataSource1",
            "isGroup": false,
            "limits": [
                {
                    "id": 1,
                    "value": 1000
                },
                ...
            ]
        },
        ...
    ]
}
```
Sample Server Failure Response

```
{
  "error": {
    "code": 222207028,
    "message": {
      "lang": "en-US",
      "value": "Missing 'userId' in payload."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the Limits (27) permission.

Get a limit set on a data source

Purpose

Retrieves a limit that has been set on a data source.

URL

https://<myserver>:<port>/api/admin/limits/users/{userId}/datasources/{dataSourceId}/{limitId}

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account to which the data source belongs.</td>
<td>The user ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The data source ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a data source.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format.

```json
{
    "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```
Status code: 200
Successful response

{
    "value": 500
}
```

Sample Server Failure Response

```
{
    "error": {
        "code": 222207004,
        "message": {
            "lang": "en-US",
            "value": "There is no DataSource with that id: 1234."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the Limits (27) permission.
Create a limit on a data source

Purpose
Creates a limit on a data source.

URL
https://<myserver>:<port>/api/admin/limits/users/{userId}/datasources/{datasourceId}/{limitId}

Method
POST

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account to which the data source belongs.</td>
<td>The user ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The data source ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a data source.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request takes the following format.

```json
{
  "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>
Sample Request Payload

```
{
  "value": 500
}
```

Sample Server Success Response

```
Status code: 201
Successful response

{
  "value": 500
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222207929,
    "message": {
      "lang": "en-US",
      "value": "Datasource with id={0} does not belong to user with id={1}"  
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the Limits (27) permission.

Update a limit on a data source

Purpose

Updates a limit that has been set on a data source.

URL

```
https://<myserver>:<port>/api/admin/limits/users/{userId}/datasources/{dataSourceId}/{limitId}
```

Method

PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account to which the data source belongs.</td>
<td>The user ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{dataSourceId}</td>
<td>The ID of the data source.</td>
<td>The data source ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a data source.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request takes the following format.

```json
{
    "value": limit_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the limit.</td>
<td>An integer that meets the requirements of the minimum and maximum values for the limit.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```json
{
    "value": 500
}
```

Sample Server Success Response

```
Status code: 200
Successful response

{
    "value": 500
}
```

Sample Server Failure Response

```
{  
    "error": {
        "code": 222207926,
        "message": {
            "lang": "en-US",
            "value": "Datasource with id={0} does not belong to user with id={1}"
        }
    }
}
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the Limits (27) permission.

Delete a limit on a data source

Purpose
Deletes a limit that has been set on a data source.

URL
https://<myserver>:<port>/api/admin/limits/users/{userId}/datasources/{datasourceId}/{limitId}

Method
DELETE

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The following parameters are also required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{userId}</td>
<td>The ID of the user account to which the data source belongs.</td>
<td>The user ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The data source ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{limitId}</td>
<td>The ID of the limit.</td>
<td>MaxRowFetchSize (1) and ODataMaxConcurrentQueries (6) are the only limits that can be set on a data source.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

Sample Server Failure Response

```json
{
  "error": {
```
"code": 222207926,
"message": {
  "lang": "en-US",
  "value": "Datasource with id={0} does not belong to user with id={1}"
}
}

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) or the Limits (27) permission.

**Logging API**

Hybrid Data Pipeline provides data source logging to record user activity against data sources. Administrators can view and set logging levels for data sources through the Logging API. The resulting data source activity log can be used to troubleshoot issues. See Data source logging on page 176 for more information.

**Note:** Enabling and increasing logging levels may adversely impact performance. Therefore, best practices recommend that logging levels be restored to their defaults once an issue has been resolved.

The following table summarizes Logging API operations.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve the logging levels for a data source</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{userid}/datasources/{datasourceid}/logging</td>
</tr>
<tr>
<td>Update the logging levels for a data source</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{userid}/datasources/{datasourceid}/logging</td>
</tr>
</tbody>
</table>

**Get logging levels for a data source**

**Purpose**

Retrieves the logging levels for a data source.

**URL**

https://<myserver>:<port>/api/admin/users/{userid}/datasources/{datasourceid}/logging

**Method**

GET
### URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameters "userid" and "datasourceid" described in the following table are required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userid&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;datasourceid&quot;</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

### Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "dasLogLevel": "logging_level",
    "privacyLevel": "privacy_level",
    "driverLogConfig": [
        {
            "name": "ADAPTER",
            "logLevel": "adapter_level"
        },
        {
            "name": "CLOUD",
            "logLevel": "cloud_level"
        },
        {
            "name": "DRIVERCOMMUNICATION",
            "logLevel": "drivercom_level"
        },
        {
            "name": "SQL",
            "logLevel": "sql_level"
        }
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;dasLogLevel&quot;</td>
<td>Determines the level of detail to be included in the data source activity log.</td>
<td>See Setting data source logging levels on page 177.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;privacyLevel&quot;</td>
<td>Determines the type of information that gets logged.</td>
<td>See Setting data source logging levels on page 177.</td>
</tr>
<tr>
<td>&quot;driverLogConfig&quot;</td>
<td>Driver loggers available for non-relational data sources and the corresponding setting for each. When these loggers are enabled, information related to the internal SQL engine is passed to the data source activity log.</td>
<td>See Setting data source logging levels on page 177.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```
Status code: 200
Successful response

{
  "dasLogLevel": "CONFIG",
  "privacyLevel": "AllowNone",
  "driverLogConfig": [
    {
      "name": "ADAPTER",
      "logLevel": "OFF"
    },
    {
      "name": "CLOUD",
      "logLevel": "OFF"
    },
    {
      "name": "DRIVERCOMMUNICATION",
      "logLevel": "OFF"
    },
    {
      "name": "SQL",
      "logLevel": "OFF"
    }
  ]
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222207004,
    "message": {
      "lang": "en-US",
      "value": "There is no DataSource with that id: 1234."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password
Authorization
The user must have the Administrator (12) permission; or the user must have the Logging (24) permission and administrative access on the tenant to which the users and data sources belong.

Update logging levels for a data source

Purpose
Updates the logging levels for a data source.

URL
https://<myserver>:<port>/api/admin/users/{userid}/datasources/{datasourceid}/logging

Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameters "userid" and "datasourceid" described in the following table are required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userid&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;datasourceid&quot;</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "dasLogLevel": "logging_level",
    "privacyLevel": "privacy_level",
    "driverLogConfig": [
    {
        "name": "ADAPTER",
        "logLevel": "adapter_level"
    },
    {
        "name": "CLOUD",
        "logLevel": "cloud_level"
    },
    {
        "name": "DRIVERCOMMUNICATION",
        "logLevel": "drivercom_level"
    },
    {
        "name": "SQL",
```
"logLevel": "sql_level"

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;dasLogLevel&quot;</td>
<td>Determines the level of detail to be included in the data source activity log.</td>
<td>See Setting data source logging levels on page 177.</td>
</tr>
<tr>
<td>&quot;privacyLevel&quot;</td>
<td>Determines the type of information that gets logged.</td>
<td>See Setting data source logging levels on page 177.</td>
</tr>
<tr>
<td>&quot;driverLogConfig&quot;</td>
<td>Driver loggers available for non-relational data sources and the corresponding setting for each. When these loggers are enabled, information related to the internal SQL engine is passed to the data source activity log.</td>
<td>See Setting data source logging levels on page 177.</td>
</tr>
</tbody>
</table>

Sample Payload Request

```json
{
   "dasLogLevel": "CONFIG",
   "privacyLevel": "AllowSQL",
   "driverLogConfig": [
   {
     "name": "ADAPTER",
     "logLevel": "SEVERE"
   },
   {
     "name": "CLOUD",
     "logLevel": "SEVERE"
   },
   {
     "name": "DRIVERCOMMUNICATION",
     "logLevel": "SEVERE"
   },
   {
     "name": "SQL",
     "logLevel": "SEVERE"
   }
   ]
}
```

Sample Server Success Response

Status code: 200
Successful response

```json
{
   "dasLogLevel": "CONFIG",
   "privacyLevel": "AllowSQL",
   "driverLogConfig": [
   {
     "name": "ADAPTER",
```
Sample Server Failure Response

```json
{
  "error":{
    "code":222207936,
    "message":{
      "lang":"en-US",
      "value":"Invalid Driver Logger name: abc. Allowed Values are adapter, sql, drivercommunication, cloud (case insensitive)."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission; or the user must have the Logging (24) permission and administrative access on the tenant to which the users and data sources belong.

Roles API

Hybrid Data Pipeline user accounts must have at least one assigned role. A role is defined by the permissions that are associated with it. The Roles API can be used to create, view, modify, and delete roles, and, more generally, manage roles and the users associated with them.

**Note:** The system administrator, tenant administrator, and user roles are predefined. These roles cannot be deleted, and only the users associated with them via the "users" property can be modified. Other properties, such as "name" and "permissions," cannot be modified.

In a single-tenant environment, all roles belong to the default system tenant. In a multitenant environment, roles must belong to specific tenants. One role cannot be used across multiple tenants. When creating a new tenant using the Tenant API, roles in the system tenant can be imported to the new tenant. The imported role is given its own ID and can only be assigned to users in the new tenant.

Any user with the Administrator (12) permission is in effect a system administrator. System administrators can create, view, modify, and delete roles in all tenants across the system. In contrast, administrator users who do not have the Administrator permission must be granted permissions for specific operations and administrative access on the tenant which they are administering.
The Roles API can be used to perform the operations described in the following table.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns list of available roles</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/roles</td>
</tr>
<tr>
<td>Creates a new role</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/roles</td>
</tr>
<tr>
<td>Returns details on the specified role</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/roles/{id}</td>
</tr>
<tr>
<td>Updates the specified role</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/roles/{id}</td>
</tr>
<tr>
<td>Deletes the specified role</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/roles/{id}</td>
</tr>
</tbody>
</table>

**Get roles**

**Purpose**

Returns list of available roles

**URL**

https://<myserver>:<port>/api/admin/roles

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Definition**

The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
    "roles": [
        {
            "id": role_id,
            "name": "role_name",
            "tenantId": tenant_id,
            "description": "role_description"
        },
        ...
    ]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the role.</td>
<td>The ID of a predefined role, such as a system administrator, or the ID of a role created by an administrator. The ID of a role cannot be changed.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the role.</td>
<td>System Administrator</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the role belongs.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the role.</td>
<td>System Administrator role has all permissions. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified. User role has all permissions associated with a user who might query data sources directly. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified. Tenant Administrator role has user permissions and permissions associated with provisioning users. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified. Optionally, administrators can provide a description for any roles they create.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

```
Status code: 200
Successful response

{
  "roles": [
    {
      "id": 1,
      "name": "System Administrator",
      "tenantId": 1,
      "description": "This role has all permissions. This role cannot be modified or deleted."
    },
    {
      "id": 2,
      "name": "User",
      "tenantId": 1,
    }
  ]
}
```
"description": "This role has the default permissions that a normal user will be expected to have."
},
{
   "id": 3,
   "name": "Tenant Administrator",
   "tenantId": 1,
   "description": "This role has all the tenant administrator permissions."
},
{
   "id": 72,
   "name": "User",
   "tenantId": 57,
   "description": "This role has the default permissions that a normal user will be expected to have."
},
{
   "id": 73,
   "name": "Tenant Administrator",
   "tenantId": 57,
   "description": "This role has all the tenant administrator permissions."
}
]

Sample Server Failure Response

{
   "error":{
      "code":222207919,
      "message":{
         "lang":"en-US",
         "value":"Problem getting Roles at this time. Please try again at another time."
      }
   }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the ViewRole (18) permission and administrative access on the tenant.

Create a role

Purpose

Creates a new role

URL

https://<myserver>:<port>/api/admin/roles

Method

POST
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Definition

The request payload is a JSON object defined as follows:

```json
{
   "name": "role_name",
   "tenantId": tenant_id,
   "description": "role_description",
   "permissions": [permission_id,permission_id,...],
   "users": [user_id,user_id,...]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the role.</td>
<td>Required</td>
<td>System Administrator</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>custom_role is the name of a role created by an administrator.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the role belongs. If not specified, the role is created in the tenant to which the user belongs.</td>
<td>Optional</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the role.</td>
<td>Optional</td>
<td>System Administrator role has all permissions. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>User role has all permissions associated with a user who might query data sources directly. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Tenant Administrator role has user permissions and permissions associated with provisioning users. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Optionally, administrators can provide a description for any roles they create.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------</td>
<td>--------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated with the role.</td>
<td>Required</td>
<td>A comma-separated list of permission IDs. See Administrator Permissions API on page 1039 for details. While this property must be included in the request payload, an empty array can be passed.</td>
</tr>
<tr>
<td>&quot;users&quot;</td>
<td>A list of users granted the role.</td>
<td>Required</td>
<td>A comma-separated list of user IDs. Note: The users property must be included in the payload, but an empty array can be passed.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```json
{
    "name": "Reader",
    "tenantId": 56,
    "description": "This role allows read-only access.",
    "permissions": [
        2,
        5,
        6,
        7
    ],
    "users": []
}
```

**Sample Server Success Response**

A successful server response will include an auto-generated ID for the newly created role.

```
Status code: 201
Successful response

{
    "id": 29,
    "name": "Reader",
    "tenantId": 56,
    "description": "This role allows read-only access.",
    "permissions": [
        2,
        5,
        6,
        7
    ],
    "users": []
}
```

**Sample Server Failure Response**

```json
{
    "error": {
        "code": 222207917,
        "message": {
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the CreateRole (17) permission and administrative access on the tenant.

Get details on a role

Purpose
Returns details on a role

URL
https://<myserver>:<port>/api/admin/roles/{id}

Method
GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the role.</td>
<td>The ID of a predefined role, such as a system administrator, or the ID of a role created by an administrator. The ID of a role cannot be changed.</td>
</tr>
</tbody>
</table>

Response Definition
The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
    "id": role_id,
    "name": "role_name",
    "tenantId": tenant_id,
    "description": "role_description",
    "permissions": [permission_id,permission_id,...]
}
```
"users": [user_id, user_id,...]

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the role.</td>
<td>The ID of a predefined role, such as a system administrator, or the ID of a role created by an administrator. The ID of a role cannot be changed.</td>
</tr>
</tbody>
</table>
| "name"       | The name of the role.        | System Administrator | User | Tenant Administrator | custom_role
                  | custom_role is the name of a role created by an administrator. |
| "tenantId"   | The ID of the tenant to which the role belongs. | A valid tenant ID. |
| "description"| The description of the role. | System Administrator role has all permissions. This role cannot be deleted, and only the users associated with it via the "users" property can be modified. Other properties, such as "name" and "permissions," cannot be modified. User role has all permissions associated with a user who might query data sources directly. This role cannot be deleted, and only the users associated with it via the "users" property can be modified. Other properties, such as "name" and "permissions," cannot be modified. Tenant Administrator role has user permissions and permissions associated with provisioning users. This role cannot be deleted, and only the users associated with it via the "users" property can be modified. Other properties, such as "name" and "permissions," cannot be modified. Optionally, administrators can provide a description for any roles they create. |
| "permissions"| A list of permissions associated with the role. | A comma-separated list of permission IDs. See Administrator Permissions API on page 1039 for details. |
| "users"      | A list of users granted the role. | A comma-separated list of user IDs. |

Sample Server Success Response

Status code: 200
Successful response
Chapter 10: Hybrid Data Pipeline API reference

```json
{
    "id": 29,
    "name": "Reader",
    "tenantId": 56,
    "description": "This role allows read-only access.",
    "permissions": [2, 5, 6, 7],
    "users": []
}
```

**Sample Server Failure Response**

```json
{
    "error":{
        "code":222207924,
        "message":{
            "lang":"en-US",
            "value":"There is no Role with that id: 1234"
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission, or the ViewRole (18) permission and administrative access on the tenant.

**Update a role**

**Purpose**

Updates the specified role

**Note:** System Administrator, User, and Tenant Administrator roles are predefined. These roles cannot be deleted, and only the users associated with them via the "users" property can be modified. Other properties, such as "name" and "permissions," cannot be modified.

**URL**

https://<myserver>:<port>/api/admin/roles/{id}

**Method**

PUT
URL Parameters

<myserv> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the role.</td>
<td>The ID of a predefined role, such as a system administrator, or the ID of a role created by an administrator. The ID of a role cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request payload is a JSON object defined as follows:

```
{
  "name": "role_name",
  "tenantId": tenant_id,
  "description": "role_description",
  "permissions": [permission_id,permission_id,...],
  "users": [user_id,user_id,...]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the role.</td>
<td>Required</td>
<td>System Administrator</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>custom_role is the name of a role created by an administrator.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the role belongs. If not specified, it is assumed the role belongs to the user's tenant.</td>
<td>Optional</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the role.</td>
<td>Optional</td>
<td>System Administrator role has all permissions. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified. User role has all permissions associated with a user who might query data sources directly. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified. Tenant Administrator role has user permissions and permissions associated with provisioning users. This role cannot be deleted, and only the users associated with it via the &quot;users&quot; property can be modified. Other properties, such as &quot;name&quot; and &quot;permissions,&quot; cannot be modified. Optionally, administrators can provide a description for any roles they create.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated with the role</td>
<td>Required</td>
<td>A comma-separated list of permission IDs. See Administrator Permissions API on page 1039 for details.</td>
</tr>
<tr>
<td>&quot;users&quot;</td>
<td>A list of users granted the role</td>
<td>Required</td>
<td>A comma-separated list of user IDs.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```
{
    "name": "Reader",
    "tenantId": 56,
    "description": "This role allows read-only access.",
    "permissions": [
        2,
        5,
        6
    ],
    "users": []
}
```

Sample Server Success Response

```
Status code: 200
Successful response

{
    "id": 29,
    "name": "Reader",
    "tenantId": 56,
    "description": "This role allows read-only access."
}
```
"permissions": [2, 5, 6],
"users": []
}

Sample Server Failure Response
{
  "error": {
    "code": 222207916,
    "message": {
      "lang": "en-US",
      "value": "There is no User with that id: 1234."
    }
  }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the ModifyRole (19) permission and administrative access on the tenant.

Delete a role

Purpose
Deletes the specified role. A role cannot be deleted if there are any users assigned to it.

Note: System Administrator, User, and Tenant Administrator roles are predefined. These roles cannot be deleted.

URL
https://<myserver>:<port>/api/admin/roles/{id}

Method
DELETE

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The URL parameter \{id\} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the role.</td>
<td>The ID of a predefined role, such as a system administrator, or the ID of a role created by an administrator. The ID of a role cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

```
{
  "success":true
}
```

Sample Server Failure Response

```
{
  "error":{
    "code":222207924,
    "message":{
      "lang":"en-US",
      "value":"There is no Role with that id: 1234"
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the DeleteRole (20) permission and administrative access on the tenant.

System Configurations API

The System Configurations API can be used for the following purposes:

- To set a delimiter for an authentication service (see also Advanced options for authentication for details)
- To set change password functionality
- To set the default OData version for data sources
- To set the default entity name mode for OData Version 4 data sources
- To enable or disable the third party JDBC data store plugin feature
- To enable or disable the default password policy
- To configure how the system persists system monitor details
- To enable or disable the IP whitelist feature
The System Configurations API supports the operations described in the following table.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve system configurations</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/configurations</td>
</tr>
<tr>
<td>Retrieve information on a system configuration</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/configurations/&lt;ID&gt;</td>
</tr>
<tr>
<td>Update a system configuration</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/configurations/&lt;ID&gt;</td>
</tr>
</tbody>
</table>

**Get Configurations**

**Purpose**

Returns a list of system configuration settings.

**URL**

https://<myserver>:<port>/api/admin/configurations

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Definition**

The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
    "configurations": [
        {
            "id": "attribute_id",
            "description": "attribute_description",
            "value": "attribute_value"
        }
    ]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the configurations attribute being returned.</td>
<td>1</td>
</tr>
</tbody>
</table>
|          |             | 1 is the ID for setting the delimiter for an authentication service.  
|          |             | 2 is the ID for secureChangePassword.  
|          |             | 3 is the ID for setting the default OData version for new data sources.  
|          |             | 4 is the ID for setting the default entity name mode for OData V4 data sources.  
|          |             | 5 is the ID for enabling or disabling third party JDBC data store plugin feature.  
|          |             | 6 is the ID for enabling or disabling the default Password Policy.  
|          |             | 7 is the ID to configure how the system persists system monitor details.  
|          |             | 8 is the ID to configure the IP whitelist filtering feature.  |
| "description" | The description of the configurations attribute. | See sample response below. |
| "value" | The value of the configurations attribute. | See sample response below. |

Sample Server Success Response

Status code: 200
Successful response

```
{
  "configurations": [
    {
      "id": 1,
      "description": "Delimiter between user name and authentication service/configuration name",
      "value": null
    },
    {
      "id": 2,
      "description": "Enable Secure Password Change, when value is set to true, the change password api will require a valid old password in order to update the logged in user password.",
      "value": "true"
    },
    {
      "id": 3,
      "description": "Default OData version for new data sources. Valid values are 2 or 4.",
      "value": "4"
    },
    {
      "id": 4,
      "description": "Default entity name mode for OData V4 data sources. Valid values are: GUESS, PLURALIZE, SINGULARIZE and SUFFIX",
      "value": "GUESS"
    }
  ]
}
```
{  "id": 5,
    "description": "Disable or enable third party JDBC data store. When the value is set to true, third party JDBC data store will be enabled. When the value is set to false, third party JDBC data store will be disabled. By default, this is set to 'true'.",
    "value": "true"
},
{  "id": 6,
    "description": "Valid values are: 1 or -1. Value of 1 enforces that the password be in compliant with the default password policy. Value of -1 turns off the Password Policy enforcement. Any other value will be treated like -1",
    "value": "-1"
},
{  "id": 7,
    "description": "Configures how the system persists system monitor details. 0 - no persistence, 1 - (default) log, 2 - database, 3 - log and database",
    "value": "1"
},
{  "id": 8,
    "description": "Configure whitelist filtering. Enables filtering when value is set to 'true'. Default value is "true"",
    "value": "true"
}

Sample Server Failure Response
{
    "error": {
        "code": 222206007,
        "message": {
            "lang": "en-US",
            "value": "Invalid user ID or password."
        }
    }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the Configurations (22) permission.

Get Configuration for given ID

Purpose
Returns the configuration settings for a given ID.

URL
https://<myserver>:<port>/api/admin/configurations/{id}
**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the configurations attribute being returned.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 is the ID for setting the delimiter for an authentication service.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 is the ID for secureChangePassword.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 is the ID for setting the default OData version for new data sources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 is the ID for setting the default entity name mode for OData V4 data sources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 is the ID for enabling or disabling third party JDBC data store plugin feature.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 is the ID for enabling or disabling the default Password Policy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 is the ID to configure how the system persists system monitor details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 is the ID to configure the IP whitelist filtering feature.</td>
</tr>
</tbody>
</table>

**Response Definition**

The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
    "id": attribute_id,
    "description": "attribute_description",
    "value": "attribute_value"
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the configurations attribute being returned.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 is the ID for setting the delimiter for an authentication service.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 is the ID for secureChangePassword.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 is the ID for setting the default OData version for new data sources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 is the ID for setting the default entity name mode for OData V4 data sources.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 is the ID for enabling or disabling third party JDBC data store plugin feature.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 is the ID for enabling or disabling the default Password Policy.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 is the ID to configure how the system persists system monitor details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 is the ID to configure the IP whitelist filtering feature.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The description of the configurations attribute.</td>
<td>For values, see the sample response in Gets configuration.</td>
</tr>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the configurations attribute.</td>
<td>For values, see the sample response in Gets configuration.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

A sample successful response has the format:

```
Status code: 200
Successful response

{
  "id": 1,
  "description": "Delimiter between user name and authentication service/configuration name",
  "value": null
}
```

**Sample Server Failure Response**

```
{
  "error": {
    "code": 222206007,
    "message": {
      "lang": "en-US",
      "value": "Invalid user ID or password."
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password
Authorization
The user must have the Administrator (12) or the Configurations (22) permission.

Update Configuration for given ID

Purpose
Updates a system configuration setting.

URL
https://<myserver>:<port>/api/admin/configurations/{id}

Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the configurations attribute being returned.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1 is the ID for setting the delimiter for an authentication service.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 is the ID for secureChangePassword.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 is the ID for setting the default OData version for new data sources.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 is the ID for setting the default entity name mode for OData V4 data sources.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 is the ID for enabling or disabling third party JDBC data store plugin feature.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 is the ID for enabling or disabling the default Password Policy.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7 is the ID to configure how the system persists system monitor details.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 is the ID to configure the IP whitelist filtering feature.</td>
<td></td>
</tr>
</tbody>
</table>
**Request Payload Definition**

The request takes the following format. The request includes the property described in the table that follows.

```json
{
   "value": attribute_value
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;value&quot;</td>
<td>The value of the configurations attribute.</td>
<td>Valid values vary depending on the attribute. For an authentication delimiter (1), a string can be specified. It is recommended to set a single character that is not generally used in a service name. (for example, &quot;:&quot; or &quot;:&quot;). By default, the value is null. For secureChangePassword (2), true is specified to require the user to specify a current password as well as a new password. False is specified to require only a new password. For default OData version for new data sources (3), valid values are 2 or 4. For default entity mode for OData V4 sources (4), valid values are: GUESS, PLURALIZE, SINGULARIZE and SUFFIX*. For Enable JDBC data store (5), valid values are true and false. When value is set to true, JDBC data store will be enabled. For default password policy (6), valid values are: 1 or -1. Value of 1 enforces that the password be in compliant with the default password policy. A value of -1 turns off the Password Policy enforcement. Note that any other value will be treated as -1. For System monitor details persistence (7), valid values are: 0 - no persistence, 1 - log, 2 - database, 3 - log and database. By default, the values is 1. For IP whitelist filtering (8), valid values are &quot;true&quot; and &quot;false&quot;. By default, the value is &quot;true&quot;.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

The following PUT operation sets the external authentication delimiter to the bar symbol (|).

```
https://MyServer:8443/api/admin/configurations/1
```

```json
{
   "value": "|
```

Sample Server Failure Response

```json
{
    "error": {
        "code": 222206007,
        "message": {
            "lang": "en-US",
            "value": "Invalid user ID or password."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the Administrator (12) or the Configurations (22) permission.

## Tenant API

The Tenant API allows administrators to create, view, modify, and delete tenants. To use the Tenant API, a user must have either the Administrator (12) permission or the TenantAPI (25) permission. Any user with the Administrator permission is in effect a system administrator. System administrators can create tenants and can view, modify, and delete all tenants across the system. Users must have the TenantAPI permission to create tenants. Users must also have administrative access for a given tenant to be able to view, modify, and delete tenants. There are two ways users may obtain administrative access on specific tenants. First, users have administrative access on any tenant they create. Second, a user can be granted administrative access on a tenant when the tenant is created or by updating the list of administrators on a tenant.

The Tenant API can be used to perform the following operations.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve a list of tenants in the system</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/tenants</td>
</tr>
<tr>
<td>Create a new tenant</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/tenants</td>
</tr>
<tr>
<td>Retrieve information on a tenant</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/tenants/{id}</td>
</tr>
<tr>
<td>Update a tenant</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/tenants/{id}</td>
</tr>
<tr>
<td>Delete a tenant</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/tenants/{id}</td>
</tr>
<tr>
<td>Retrieve the list of administrators for a tenant</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/tenants/{id}/admins</td>
</tr>
<tr>
<td>Update the list of administrators for a tenant</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/tenants/{id}/admins</td>
</tr>
</tbody>
</table>
Get tenants

Purpose
Returns a list of all tenants

URL
https://<myserver>:<port>/api/admin/tenants

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition
The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
   "tenants": [
   {
      "id": tenant_id,
      "name": "tenant_name"
   },
   ...
]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the tenant.</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```json
{
   "tenants": [
   {
      "id": 1,
      "name": "System"
   },
   {
      "id": 71,
      "name": "OrgA"
   }
]
```
Sample Server Failure Response

```json
{
  "error": {
    "code": 222208103,
    "message": {
      "lang": "en-US",
      "value": "You lack the permissions to access this url."
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) or the TenantAPI (25) permission.

**Create a tenant**

**Purpose**

Creates a tenant

**URL**

https://<myserver>:<port>/api/admin/tenants

**Method**

POST

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
Request Payload Definition
The request payload is a JSON object defined as follows:

```
{
   "name": "tenant_name",
   "description": "tenant_description",
   "parentTenant": 1,
   "status": 1,
   "importedRoles": [role_id, role_id, ...],
   "admins": [56]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the tenant.</td>
<td>Required</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the tenant.</td>
<td>Optional</td>
<td>A string that provides a description of the tenant.</td>
</tr>
<tr>
<td>&quot;parentTenant&quot;</td>
<td>The ID of the parent tenant.</td>
<td>Required</td>
<td>The system tenant is currently the only tenant that can act as a parent tenant. Therefore, the only valid value is 1, the ID of the system tenant.</td>
</tr>
<tr>
<td>&quot;status&quot;</td>
<td>The status of the tenant.</td>
<td>na</td>
<td>This option will be available with a future update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 specifies that the tenant is inactive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 specifies that the tenant is active.</td>
</tr>
</tbody>
</table>
A valid role ID or comma-separated list of valid role IDs in the parent tenant. These roles are copied to the new tenant and given their own unique IDs.

**Note:** Any role, including the system administrator role, with the Administrator (12) permission cannot be copied to a tenant.

A list of administrators who have administrative access to the tenant. Any user that appears in this list has administrative access on the tenant. However, the user must have permissions to execute corresponding operations. When creating a tenant, any administrator users listed must reside in the system tenant. After the tenant has been created, users provisioned within the tenant can be granted administrative access.

### Sample Request Payload

```json
{
    "name": "OrgB",
    "description": "This is the HDP tenant for organization B.",
    "parentTenant": 1,
    "status": 1,
    "importedRoles": [
        2,
        3
    ],
    "admins": [
        2
    ]
}
```

### Sample Server Success Response

A successful server response will include an auto-generated ID for the newly created tenant. The imported roles will also be given their own unique IDs.

Status code: 201
Successful response

```json
{
    "id": 360,
    "name": "OrgB",
    "description": "This is the HDP tenant for organization B.",
}
```
Sample Server Failure Response

```
{
    "error":{
        "code":222207917,
        "message":{
            "lang":"en-US",
            "value":"Problem creating a Role at this time. Please try again at another
time."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission or the TenantAPI (25) permission.

Get information on a tenant

Purpose

Returns information for a tenant

URL

https://<myserver>:<port>/api/admin/tenants/{id}

Additional information, including tenant roles and administrators, can be retrieved by setting the details query parameter to true (?details=true).

https://<myserver>:<port>/api/admin/tenants/{id}?details=true

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The URL parameter \{id\} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

**Response Definition**

The response takes the following format. The parameters of the response are described in the table that follows.

**Note:** The roles and admins properties are provided when the query ?details=true has been added to the URL.

```
{
    "id": tenant_id,
    "name": "tenant_name",
    "description": "tenant_description",
    "parentTenant": parent_tenant_id,
    "status": tenant_status,
    "roles": [
        role_id,
        role_id,
        ...
    ],
    "admins": [
        user_id,
        user_id,
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the tenant.</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the tenant.</td>
<td>A string that provides a description of the tenant.</td>
</tr>
<tr>
<td>&quot;parentTenant&quot;</td>
<td>The ID of the parent tenant.</td>
<td>null</td>
</tr>
<tr>
<td></td>
<td>null is returned when the query is executed for the system tenant.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 is returned when the query is executed for tenants in the system tenant.</td>
<td></td>
</tr>
<tr>
<td>&quot;status&quot;</td>
<td>The status of the tenant.</td>
<td>This option will be available with a future update.</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0 specifies that the tenant is inactive.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 specifies that the tenant is active.</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>&quot;roles&quot;</td>
<td>The role or roles that belong to the tenant.</td>
<td>A valid role ID or comma-separated list of valid role IDs.</td>
</tr>
<tr>
<td>&quot;admins&quot;</td>
<td>A list of administrators who have administrative access to the tenant.</td>
<td>A valid user ID or comma-separated list of valid user IDs. Any user that appears in this list has administrative access on the tenant. However, the user must have permissions to execute corresponding operations.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

```
Status code: 200
Successful response

{
    "id": 360,
    "name": "OrgB",
    "description": "This is the HDP tenant for organization B.",
    "parentTenant": 1,
    "status": 1
}
```

**Sample Server Failure Response**

```
{
    "error": {
        "code": 222208573,
        "message": {
            "lang": "en-US",
            "value": "There is no Tenant with that id: 22."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) or the TenantAPI (25) permission.

**Update a tenant**

**Purpose**

Updates a tenant

**URL**

https://<myserver>:<port>/api/admin/tenants/{id}
Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request payload is a JSON object defined as follows:

```json
{
    "name": "tenant_name",
    "description": "tenant_description",
    "parentTenant": parent_tenant_id,
    "status": tenant_status
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the tenant.</td>
<td>Required</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the tenant.</td>
<td>Optional</td>
<td>A string that provides a description of the tenant.</td>
</tr>
</tbody>
</table>
| "parentTenant"    | The ID of the parent tenant. | Optional                  | null|1  
null is returned when the query is executed for the system tenant.  
1 is returned when the query is executed for tenants in the system tenant. |
| "status"          | The status of the tenant. | Required                  | This option will be available with a future update.  
0|1  
0 specifies that the tenant is inactive.  
1 specifies that the tenant is active. |
Sample Request Payload

{
    "name": "OrgB",
    "description": "This is a new description.",
    "parentTenant": 1,
    "status": 1
}

Sample Server Success Response

Status code: 200
Successful response

{
    "id": 360,
    "name": "OrgB",
    "description": "This is a new description.",
    "parentTenant": 1,
    "status": 1
}

Sample Server Failure Response

{
    "error": {
        "code": 222208573,
        "message": {
            "lang": "en-US",
            "value": "There is no Tenant with that id: 22."
        }
    }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the TenantAPI (25) permission.

Delete a tenant

Purpose

Deletes a tenant

URL

https://<myserver>:<port>/api/admin/tenants/{id}

Method

DELETE
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

```json
{
   "success":true
}
```

Sample Server Failure Response

```json
{
   "error": {
      "code": 222208573,
      "message": {
         "lang": "en-US",
         "value": "There is no Tenant with that id: 22."
      }
   }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the TenantAPI (25) permission.

Get the list of administrators for a tenant

Purpose

Returns administrators for a tenant

URL

https://<myserver>:<port>/api/admin/tenants/{id}/admins

Method

GET
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format. The parameters of the response are described in the table that follows.

```
{
    "admins": [
        user_id,
        user_id,
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;admins&quot;</td>
<td>A list of administrators who have administrative access to the tenant</td>
<td>A valid user ID or comma-separated list of valid user IDs. Any user that appears in this list has administrative access on the tenant. However, the user must have permissions to execute corresponding operations.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```
Status code: 200
Successful response

{
    "admins": [
        33,
        66,
        99,
        132
    ]
}
```

Sample Server Failure Response

```
{
    "error": {
        "code": 222208573,
        "message": {
            "lang": "en-US",
            "message": "An error occurred."
        }
    }
}
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) or the TenantAPI (25) permission.

Update the list of administrators on a tenant

Purpose
Updates administrators for a tenant

URL
https://<myserver>:<port>/api/admin/tenants/{id}/admins

Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request payload is a JSON object defined as follows:

```json
{
    "admins": [
        user_id,
        user_id,
        ...
```
Valid Values Description Property

"admins"  A list of administrators who have administrative access to the tenant  A valid user ID or comma-separated list of valid user IDs.
Any user that appears in this list has administrative access on the tenant. However, the user must have permissions to execute corresponding operations.
Any administrator users listed must reside in the system tenant or the tenant that is being updated.

Sample Request Payload

```json
{
  "admins": [45, 75, 105]
}
```

Sample Server Success Response

```
Status code: 200
Successful response

{
  "admins": [45, 75, 105]
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222208573,
    "message": {
      "lang": "en-US",
      "value": "There is no Tenant with that id: 22."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) or the TenantAPI (25) permission.
Users API

The Users API can be used to provision and manage Hybrid Data Pipeline user accounts. Administrators can also use the Users API to set permissions on user accounts, assign roles to user accounts, and configure authentication for user accounts.

When working with Hybrid Data Pipeline user accounts, it is important to note that two types of authentication services are supported. First, an end user may use the default internal authentication service. In this case, the end user authenticates directly with Hybrid Data Pipeline by passing the username and password associated with a user account. Alternatively, a Hybrid Data Pipeline user account can be associated with an external authentication service. In this case, multiple end users can be associated with a single Hybrid Data Pipeline user account through the external authentication service. These end users inherit the permissions attached to the Hybrid Data Pipeline user account. (See Authentication on page 143 and Authentication API on page 1042 for details.)

Any user with the Administrator (12) permission is in effect a system administrator and has permission to perform any operation available in Hybrid Data Pipeline. They are in effect a super user. It is strongly recommended that these accounts be secured. Other administrator accounts should be created with only the permissions they need. System administrators can create, view, modify, and delete user accounts in all tenants across the system. In contrast, administrator users who do not have the Administrator permission must be granted permissions for specific operations and administrative access to the tenant which they are administering.

Note: Users in the default system tenant can be promoted to administer multiple tenants across the system. However, users in non-system tenants can only be promoted to administer users within their own tenant. They cannot administer users in other tenants.

The following table summarizes Users API operations.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve a list of user accounts</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users</td>
</tr>
<tr>
<td>Create a user account</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users</td>
</tr>
<tr>
<td>Retrieve information on a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}</td>
</tr>
<tr>
<td>Update information on a user account</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}</td>
</tr>
<tr>
<td>Delete a user account</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}</td>
</tr>
<tr>
<td>Retrieve status information on a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/statusinfo</td>
</tr>
<tr>
<td>Update status information on a user account</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/statusinfo</td>
</tr>
<tr>
<td>Retrieve password information on a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/passwordinfo</td>
</tr>
<tr>
<td>Update password information on a user account</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/passwordinfo</td>
</tr>
</tbody>
</table>
### Operation

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset the password of a user account</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/resetpassword</td>
</tr>
<tr>
<td>Retrieve permissions on a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/permissions</td>
</tr>
<tr>
<td>Update permissions on a user account</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/permissions</td>
</tr>
<tr>
<td>Get authentication information on a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/authinfo</td>
</tr>
<tr>
<td>Update authentication information on a user account</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/authinfo</td>
</tr>
<tr>
<td>Retrieve information on an authentication user</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/authUserName/{auth_user_name}</td>
</tr>
<tr>
<td>Retrieve a list of data sources for a user account</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{userid}/dataSources</td>
</tr>
<tr>
<td>Retrieve the list of tenants the user account administers</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/tenantsadministered</td>
</tr>
<tr>
<td>Update the list of tenants the user account administers</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/users/{id}/tenantsadministered</td>
</tr>
</tbody>
</table>

### Get user accounts

**Purpose**

Retrieves a list of user accounts

**URL**

https://<myserver>:<port>/api/admin/users

When the details query parameter is set to true, the response payload will include the **tenantName** property.

https://<myserver>:<port>/api/admin/users?details=true

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
## Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "users": [
        {
            "id": user_account_id,
            "userName": "user_account_name",
            "tenantId": tenant_id,
            "tenantName": "tenant_name",
            "statusInfo": {status_information},
            "passwordInfo": {password_information},
            "permissions": {permissions}
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;userName&quot;</td>
<td>The name of the user account.</td>
<td>The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the user belongs.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;tenantName&quot;</td>
<td>The name of the tenant to which the user belongs.</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Included when the details query parameter is set to true (??details=true).</td>
<td></td>
</tr>
<tr>
<td>&quot;statusInfo&quot;</td>
<td>The status of the user account defined by the status property and additional properties associated with an account lockout policy.</td>
<td>See statusInfo Object on page 1155 for details.</td>
</tr>
<tr>
<td>&quot;passwordInfo&quot;</td>
<td>Password information associated with the user account defined by the password, passwordStatus, and passwordExpiration properties.</td>
<td>See passwordInfo Object on page 1156 for details.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>Permissions associated with the user account in terms of the role(s) and permissions set explicitly on the account. User account permissions are the sum of the permissions on associated role(s) and permissions set explicitly on the account. Roles must belong to the tenant in which the user is being created.</td>
<td>See permissions Object on page 1156 for details.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

Note: The response will not return settings for optional properties that were not set in a previous POST or PUT request.

Status code: 200
Successful response

```
{
  "users": [
    {
      "id": 1,
      "userName": "d2cadmin",
      "tenantId": 1,
      "statusInfo": {
        "status": 1,
        "accountLocked": false
      },
      "permissions": {
        "roles": [
          1
        ]
      }
    },
    {
      "id": 62,
      "userName": "OrgA_Admin",
      "tenantId": 26,
      "statusInfo": {
        "status": 1,
        "accountLocked": false
      },
      "permissions": {
        "roles": [
          86
        ]
      }
    },
    {
      "id": 73,
      "userName": "OrgB_Admin",
      "tenantId": 29,
      "statusInfo": {
        "status": 1,
        "accountLocked": false
      },
      "permissions": {
        "roles": [
          94
        ]
      }
    }
  ]
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222206007,
    "message": {
      "lang": "en-US",
      "value": "Invalid user ID or password."
    }
  }
}
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

Create a user account

Purpose
Creates a user account.

URL
https://<myserver>:<port>/api/admin/users

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Properties
The request takes the following format. The properties of the request are described in the table that follows.

```
{
    "userName": "user_name",
    "tenantId": tenant_id,
    "statusInfo": {status_information},
    "passwordInfo": {password_information},
    "permissions": {permissions},
    "authenticationInfo": {authentication_information}
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userName&quot;</td>
<td>The name of the user account.</td>
<td>Required</td>
<td>The maximum length is 128 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the user belongs.</td>
<td>Optional</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> When <code>tenantId</code> is not specified, the user is created in the tenant in which the administrator executing the operation resides.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;statusInfo&quot;</td>
<td>The status of the user account defined by the <code>status</code> property and additional properties associated with an account lockout policy.</td>
<td>Required</td>
<td>See <code>statusInfo Object</code> on page 1155 for details.</td>
</tr>
<tr>
<td>&quot;passwordInfo&quot;</td>
<td>Password information associated with the user account defined by the <code>password</code>, <code>passwordStatus</code>, and <code>passwordExpiration</code> properties.</td>
<td>Optional</td>
<td>See <code>passwordInfo Object</code> on page 1156 for details.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>Permissions associated with the user account in terms of the role(s) and permissions set explicitly on the account. User account permissions are the sum of the permissions on associated role(s) and permissions set explicitly on the account. A user account may only be assigned roles in their tenant.</td>
<td>Optional</td>
<td>See <code>permissions Object</code> on page 1156 for details.</td>
</tr>
<tr>
<td>&quot;authenticationInfo&quot;</td>
<td>Authentication information associated with the user account as defined by the <code>authUserName</code> and <code>authServiceId</code> properties. The <code>authenticationInfo object</code> does not need to be included in a request payload when the default internal authentication service is being used. When an external authentication service is being used, <code>authenticationInfo</code> must be included in the request payload. If <code>authenticationInfo</code> is not passed, a default <code>authenticationInfo object</code> is created where the <code>userName</code> of the account object is used as the <code>authUserName</code> and the <code>authServiceId</code> specifies the ID of the internal authentication service (1).</td>
<td>Optional</td>
<td>See <code>authenticationInfo Object</code> on page 1157 for details.</td>
</tr>
</tbody>
</table>
Sample Payload Requests

**Example 1 payload request** The following example shows a payload request to create a user account that uses the internal authentication service. In this scenario, the end user would authenticate with the username associated with the user account (testuser).

```json
{
  "userName": "testuser",
  "tenantId": 26,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "password": "TempPassword",
    "passwordStatus": 1,
    "passwordExpiration": "2020-01-01 00:00:00"
  },
  "permissions": {
    "roles": [86]
  }
}
```

**Example 2 payload request** The following example shows a payload request to create a user account using an external authentication service. Here the end user (user_external) authenticates via an external authentication service ("authServiceId": 2). This end user inherits all the attributes associated with the testuser account.

```json
{
  "userName": "testuser",
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "password": "TempPassword",
    "passwordStatus": 1,
    "passwordExpiration": "2020-01-01 00:00:00"
  },
  "permissions": {
    "roles": [2]
  },
  "authenticationInfo": {
    "authUsers": [
      {
        "authUserName": "user_external",
        "authServiceId": 2
      }
    ]
  }
}
```

**Example 3 payload request** The following payload request creates a user account that supports both the internal authentication service or an external authentication service. The end user testuser may authenticate through the internal authentication service. Alternatively, the end user user_external can, with a distinct set of credentials, authenticate via the external authentication service "authServiceId": 2.

```json
{
  "userName": "testuser",
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "authenticationInfo": {
    "authUsers": [
      {
        "authUserName": "user_external",
        "authServiceId": 2
      }
    ]
  }
}
```
"passwordInfo": {
  "password": "TempPassword",
  "passwordStatus": 1,
  "passwordExpiration": "2020-01-01 00:00:00"
},
"permissions": {
  "roles": [2]
},
"authenticationInfo": {
  "authUsers": [
    {
      "authUserName": "user_external",
      "authServiceId": 2
    },
    {
      "authUserName": "testuser",
      "authServiceId": 1
    }
  ]
}

Sample Success Responses

Example 1 success response

Status code: 201
Successful response

{
  "id": 3,
  "userName": "testuser",
  "tenantId": 26,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "password": "TempPassword",
    "passwordStatus": 1,
    "passwordExpiration": "2020-01-01 00:00:00"
  },
  "permissions": {
    "roles": [86]
  },
  "authenticationInfo": {
    "authUsers": [
      {
        "authUserName": "testuser",
        "authServiceId": 1
      }
    ]
  }
}

Example 2 success response

Status code: 201
Successful response

{
  "id": 4,
Example 3 success response

Status code: 201
Successful response

{  
  "userName": "testuser",
  "statusInfo": {  
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {  
    "password": "TempPassword",
    "passwordStatus": 1,
    "passwordExpiration": "2020-01-01 00:00:00"
  },
  "permissions": {  
    "roles": [  
      2
    ]
  },
  "authenticationInfo": {  
    "authUsers": [  
      {  
        "authUserName": "user_external",
        "authServiceId": 2
      },
      {  
        "authUserName": "testuser",
        "authServiceId": 1
      }
    ]
  }
}

Sample Server Failure Response

{  
  "error":{  
    "code":222207415,
    "message":{  
      "lang":"en-US",
      "value":"UserName 'Joe' already exists."  
    }
  }
}
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the CreateUsers (13) permission and administrative access on the tenant.

Note: Administrator users cannot grant permissions they do not have to other user accounts.

statusInfo Object

Purpose
Describes the status information for a user account.

Syntax
```
{
    "status": integer,
    "accountLocked": boolean,
    "accountLockedAt": "YYYY-MM-DD HH:mm:ss",
    "accountLockedUntil": "YYYY-MM-DD HH:mm:ss"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;status&quot;</td>
<td>Specifies whether the user is active. An inactive user cannot log in to the</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Web UI, use APIs, or establish JDBC, ODBC, or OData connections.</td>
<td>If set to 0, the user is inactive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If set to 1, the user is active.</td>
</tr>
<tr>
<td>&quot;accountLocked&quot;</td>
<td>Specifies whether the user account has been locked based on the password</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>failure lockout policy.</td>
<td>If set to true, the account has been</td>
</tr>
<tr>
<td></td>
<td></td>
<td>locked.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If set to false, the account is not</td>
</tr>
<tr>
<td></td>
<td></td>
<td>locked.</td>
</tr>
<tr>
<td>&quot;accountLockedAt&quot;</td>
<td>Specifies the time at which the user account has been locked.</td>
<td>Timestamps must be in the UTC format</td>
</tr>
<tr>
<td></td>
<td></td>
<td>YYYY-MM-DD HH:mm:ss.</td>
</tr>
<tr>
<td>&quot;accountLockedUntil&quot;</td>
<td>Specifies the time until which the user account will remain locked.</td>
<td>Timestamps must be in the UTC format</td>
</tr>
<tr>
<td></td>
<td></td>
<td>YYYY-MM-DD HH:mm:ss.</td>
</tr>
</tbody>
</table>
passwordInfo Object

Purpose
Describes the password information for a user account.

Syntax
```
{
    "password": "string",
    "passwordStatus": integer,
    "passwordExpiration": "YYYY-MM-DD HH:mm:ss.n"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;password&quot;</td>
<td>Specifies a temporary user password. Required to support the default internal authentication service.</td>
<td>A string with a maximum length of 32 characters. The password created by the administrator is only a temporary password. Users must change the password when they log in for the first time.</td>
</tr>
</tbody>
</table>
| "passwordStatus"  | Specifies whether the password is active.                                   | 1|2  
If set to 1, the password is active.  
If set to 2, the password must be reset. |
| "passwordExpiration" | Specifies the date when the password expires.                        | Timestamps must be in the UTC format  
YYYY-MM-DD HH:mm:ss.  
If null, the password has no expiration. |

permissions Object

Purpose
Describes the permissions on the user account in terms of roles and explicitly granted permissions. The permissions on a user account are the sum of the permissions granted to the any user roles associated with the user account and permissions granted explicitly to the user account.

Syntax
```
{
    "roles": [integer, integer, ...],
    "permissions": [integer, integer, ...]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;roles&quot;</td>
<td>A role or list of roles associated with the user account. A user account must have at least one assigned role, and may only be assigned roles from its tenant.</td>
<td>The ID of the role assigned to the user account, or a comma-separated list of role IDs assigned to the user account. See also Permissions and default roles on page 59.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A permission or list of permissions granted explicitly on the user account, in addition to those based on assigned roles.</td>
<td>The ID of the permission granted, or a comma-separated list of permissions granted, to the user account. See also Permissions and default roles on page 59.</td>
</tr>
</tbody>
</table>

**authenticationInfo Object**

**Purpose**

Describes authentication information for the user account as defined by the authUserName and authServiceId properties.

The authenticationInfo object does not need to be included in a request payload when only the default internal authentication service is being used. When an external authentication service is being used, authenticationInfo must be included in the request payload. If authenticationInfo is not passed, a default authenticationInfo object is created where the userName of the system object is used as the authUserName and the authServiceId specifies the ID of the internal authentication service (1).

**Syntax**

```javascript
{
   "authUsers": [
   {
      "authUserName": "string",
      "authServiceId": integer
   },
   ...
   ]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;authUserName&quot;</td>
<td>The name of the authentication user.</td>
<td>A string where the string specifies a user name persisted by the authentication service. The maximum length is 128 characters.</td>
</tr>
</tbody>
</table>
| "authServiceId"| The ID of the authentication service against which the user is authenticating. | 1 | x  
1 is the ID for the default internal authentication service.  
X is an auto-generated ID for an external authentication service implemented by an administrator. |

**Note:** In a multitenant environment, only authentication services from the system tenant or the user’s tenant may used.

---

**Get a user account**

**Purpose**
Retrieves information on a user account

**URL**
https://<myserver>:<port>/api/admin/users/{id}

When the details query parameter is set to true, the response payload will include the tenantName and tenantsAdministered properties.

https://<myserver>:<port>/api/admin/users/{id}?details=true

**Method**
GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Response Definition

The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
    "id": user_account_id,
    "userName": "user_account_name",
    "tenantId": tenant_id,
    "tenantName": "tenant_name",
    "statusInfo": {status_information},
    "passwordInfo": {password_information},
    "permissions": {permissions},
    "authenticationInfo": {authentication_info},
    "tenantsAdministered": {tenant_id, tenant_id,...}
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;userName&quot;</td>
<td>The name of the user account.</td>
<td>The maximum length is 128 characters.</td>
</tr>
</tbody>
</table>
| "tenantId"        | The ID of the tenant to which the user belongs.                             | 1 | x  
<p>|                   | i is the ID for the system tenant.                                          |
|                   | x is the ID for a tenant created by an administrator. The ID is auto-generated when the tenant is created and cannot be changed. |
| &quot;tenantName&quot;      | The name of the tenant to which the user belongs.                           | A string that specifies the name of the tenant.                             |
|                   | <strong>Note:</strong> Included when the details query parameter is set to true (?)details=true). |
| &quot;statusInfo&quot;      | The status of the user account defined by the status property and additional properties associated with an account lockout policy. | See statusInfo Object on page 1155 for details.                             |
| &quot;passwordInfo&quot;    | Password information associated with the user account defined by the password, passwordStatus, and passwordExpiration properties. | See passwordInfo Object on page 1156 for details.                            |
| &quot;permissions&quot;     | Permissions associated with the user account in terms of the role(s) and permissions set explicitly on the account. User account permissions are the sum of the permissions on associated role(s) and permissions set explicitly on the account. | See permissions Object on page 1156 for details.                             |</p>
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;authenticationInfo&quot;</td>
<td>Authentication information associated with the user account as defined by the authUserName and authServiceId properties. The authenticationInfo object does not need to be included in a request payload when the default internal authentication service is being used. When an external authentication service is being used, authenticationInfo must be included in the request payload. If authenticationInfo is not passed, a default authenticationInfo object is created where the userName of the account object is used as the authUserName and the authServiceId specifies the ID of the internal authentication service (1).</td>
<td>See authenticationInfo Object on page 1157 for details.</td>
</tr>
<tr>
<td>&quot;tenantsAdministered&quot;</td>
<td>The ID or IDs of the tenants that the user administers.</td>
<td>A valid tenant ID or comma-separated list of valid tenant IDs.</td>
</tr>
</tbody>
</table>

**Sample Server Response**

**Note:** The response will not return settings for optional properties that were not set in a previous POST or PUT request.

```
Status code: 200
Successful response

{
  "id": 3,
  "userName": "testuser",
  "tenantId": 1,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "passwordStatus": 1,
    "passwordExpiration": null
  },
  "permissions": {
    "roles": [
      2
    ]
  },
  "authenticationInfo": {
    "authUsers": [
      {
        "authUserName": "testuser",
```
Sample Server Failure Response

```json
{
  "error": {
    "code": 222207916,
    "message": {
      "lang": "en-US",
      "value": "There is no User with that id: 123."
    }
  }
}
```

**Authentication**
Basic Authentication using Login ID and Password

**Authorization**
The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

**Update a user account**

**Purpose**
Updates information on a user account

**URL**
https://<myserver>:<port>/api/admin/users/{id}

**Method**
PUT

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>
### Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "userName": "user_name",
    "tenantId": tenant_id,
    "statusInfo": {status_information},
    "passwordInfo": {password_information},
    "permissions": {permissions},
    "authenticationInfo": {authentication_information}
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userName&quot;</td>
<td>The name of the user account</td>
<td>Required</td>
<td>The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the user belongs</td>
<td>Optional</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 is the ID for the system tenant.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>x is the ID for a tenant created by an administrator. The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;statusInfo&quot;</td>
<td>The status of the user account defined by the status property and additional properties associated with an account lockout policy.</td>
<td>Required</td>
<td>See statusInfo Object on page 1155 for details.</td>
</tr>
<tr>
<td>&quot;passwordInfo&quot;</td>
<td>Password information associated with the user account defined by the password, passwordStatus, and passwordExpiration properties.</td>
<td>Optional</td>
<td>See passwordInfo Object on page 1156 for details.</td>
</tr>
</tbody>
</table>
## Valid Values Usage Description

### Property: "permissions"

Permissions associated with the user account in terms of the role(s) and permissions set explicitly on the account. User account permissions are the sum of the permissions on associated role(s) and permissions set explicitly on the account. Roles must belong to the tenant in which the user is being created.

Usage: Optional

Valid Values: See permissions Object on page 1156 for details.

### Property: "authenticationInfo"

Authentication information associated with the user account as defined by the authUserName and authServiceId properties. The authenticationInfo object does not need to be included in a request payload when the default internal authentication service is being used. When an external authentication service is being used, authenticationInfo must be included in the request payload. If authenticationInfo is not passed, a default authenticationInfo object is created where the userName of the account object is used as the authUserName and the authServiceId specifies the ID of the internal authentication service (1).

Usage: Optional

Valid Values: See authenticationInfo Object on page 1157 for details.

---

### Sample Payload Request

**Note:** Optional properties not included in the payload request will be removed from the object.

```json
{
    "userName": "testuser",
    "tenantId": 1,
    "statusInfo": {  
        "status": 1,
        "accountLocked": false
    },
    "passwordInfo": {
        "passwordStatus": 1,
        "passwordExpiration": "2025-01-01 00:00:00"
    },
    "permissions": {
        "roles": [  
            1
        ]
    }
}
```
Sample Server Success Response

```json
{
  "userName": "testuser",
  "tenantId": 1,
  "statusInfo": {
    "status": 1,
    "accountLocked": false
  },
  "passwordInfo": {
    "passwordStatus": 1,
    "passwordExpiration": "2025-01-01 00:00:00"
  },
  "permissions": {
    "roles": [
      1
    ]
  },
  "authenticationInfo": {
    "authUsers": [
      {
        "authUserName": "testuser",
        "authServiceId": 1
      }
    ]
  }
}
```

Sample Server Failure Response

```json
{
  "error": {
    "code": 222207916,
    "message": {
      "lang": "en-US",
      "value": "There is no User with that id: 123."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.

Note: Administrator users cannot grant permissions they do not have to other user accounts.

Delete a user account

Purpose

Deletes a system user
**URL**

https://<myserver>:<port>/api/admin/users/{id}

**Method**

DELETE

**URL Parameters**
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

Status code: 204
Successful response

```json
{
  "success":true
}
```

**Sample Server Failure Response**

```json
{
  "error": {
    "code": "222207916",
    "message": {
      "lang": "en-US",
      "value": "There is no User with that id: 123."
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the Administrator (12) permission, or the DeleteUsers (16) permission and administrative access on the tenant.

**Get status info on a user account**

**Purpose**

Retrieves status information on a user account
URL
https://<myserver>:<port>/api/admin/users/{id}/statusinfo

Method
GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```
{
    "status": integer,
    "accountLocked": boolean,
    "accountLockedAt": "YYYY-MM-DD HH:mm:ss",
    "accountLockedUntil": "YYYY-MM-DD HH:mm:ss"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;status&quot;</td>
<td>Specifies whether the user is active. An inactive user cannot log in to the Web UI, use APIs, or establish JDBC, ODBC, or OData connections.</td>
<td>Required</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If set to 0, the user is inactive. If set to 1, the user is active.</td>
</tr>
<tr>
<td>&quot;accountLocked&quot;</td>
<td>Specifies whether the user account has been locked based on the password failure lockout policy.</td>
<td>Optional</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If set to true, the account has been locked. If set to false, the account is not locked.</td>
</tr>
<tr>
<td>&quot;accountLockedAt&quot;</td>
<td>Specifies the time at which the user account has been locked.</td>
<td>Optional</td>
<td>Timestamps must be in the UTC format <code>YYYY-MM-DD HH:mm:ss</code>.</td>
</tr>
<tr>
<td>&quot;accountLockedUntil&quot;</td>
<td>Specifies the time until which the user account will remain locked.</td>
<td>Optional</td>
<td>Timestamps must be in the UTC format <code>YYYY-MM-DD HH:mm:ss</code>.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

Note: The response will not return settings for optional properties that were not set in a previous POST or PUT request.

Status code: 200
Successful response

{
  "status": 1,
  "accountLocked": true,
  "accountLockedAt": "2018-02-02 05:24:12",
  "accountLockedUntil": "2018-02-02 05:54:12"
}

Sample Server Failure Response

{
  "error":{
    "code":222207916,
    "message":{
      "lang":"en-US",
      "value":"There is no User with that id: 123."
    }
  }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

Update status info on a user account

Purpose

Updates status information on a user account

URL

https://<myserver>:<port>/api/admin/users/{id}/statusinfo

Method

PUT
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
   "status": integer,
   "accountLocked": boolean,
   "accountLockedAt": "YYYY-MM-DD HH:mm:ss",
   "accountLockedUntil": "YYYY-MM-DD HH:mm:ss"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;status&quot;</td>
<td>Specifies whether the user is active.</td>
<td>Required</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>An inactive user cannot log in to the Web UI, use APIs, or establish JDBC, ODBC, or OData connections.</td>
<td></td>
<td>If set to 0, the user is inactive. If set to 1, the user is active.</td>
</tr>
<tr>
<td>&quot;accountLocked&quot;</td>
<td>Specifies whether the user account has been locked based on the password failure lockout policy.</td>
<td>Optional</td>
<td>true</td>
</tr>
<tr>
<td>&quot;accountLockedAt&quot;</td>
<td>Specifies the time at which the user account has been locked.</td>
<td>Optional</td>
<td>Timestamps must be in the UTC format YYYY-MM-DD HH:mm:ss.</td>
</tr>
<tr>
<td>&quot;accountLockedUntil&quot;</td>
<td>Specifies the time until which the user account will remain locked.</td>
<td>Optional</td>
<td>Timestamps must be in the UTC format YYYY-MM-DD HH:mm:ss.</td>
</tr>
</tbody>
</table>
Sample Request Payload

**Note:** Optional properties not included in the payload request will be removed from the object.

```json
{
    "status": 1,
    "accountLocked": false
}
```

Sample Server Response

Status code: 200
Successful response

```json
{
    "status": 1,
    "accountLocked": false
}
```

Sample Server Failure Response

```json
{
    "error":{
        "code":222207916,
        "message":{
            "lang":"en-US",
            "value":"There is no User with that id: 123."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.

Get password info on a user account

Purpose

Returns password information on a user account. This call cannot be used to retrieve the password.

URL

https://<myserver>:<port>/api/admin/users/{id}/passwordinfo

Method

GET
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```
{
    "passwordStatus": integer,
    "passwordExpiration": "YYYY-MM-DD HH:mm:ss"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;passwordStatus&quot;</td>
<td>Specifies whether the password is active.</td>
<td>Required</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If set to 1, the password is active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If set to 2, the password must be reset.</td>
</tr>
<tr>
<td>&quot;passwordExpiration&quot;</td>
<td>Specifies the date when the password expires.</td>
<td>Optional</td>
<td>Timestamps must be in the UTC format</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>YYYY-MM-DD HH:mm:ss.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If null, the password has no expiration.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Note: The response will not return settings for optional properties that were not set in a previous POST or PUT request.

```
Status code: 200
Successful response

{
    "passwordStatus": 1,
    "passwordExpiration": "2020-02-02 00:00:00"
}
```

Sample Server Failure Response

```
{
    "error":{
        "code":222207916,
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

Update password info on a user account

Purpose
Updates password information on a user account. This call cannot be used to reset the password. See Reset the password on a user account on page 1173 for information on resetting a user's password.

URL
https://<myserver>:<port>/api/admin/users/{id}/passwordinfo

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.

```json
{
   "passwordStatus": integer,
   "passwordExpiration": "YYYY-MM-DD HH:mm:ss"
}
```
### Property Table

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;passwordStatus&quot;</td>
<td>Specifies whether the password is active.</td>
<td>Required</td>
<td>1</td>
</tr>
</tbody>
</table>
| "passwordExpiration" | Specifies the date when the password expires.                  | Optional    | Timestamps must be in the UTC format YYYY-MM-DD HH:mm:ss.  
If null, the password has no expiration. |

### Sample Request Payload

**Note:** Optional properties not included in the payload request will be removed from the object.

```json
{
  "passwordStatus": 2,
  "passwordExpiration": "2025-12-31 00:00:00"
}
```

### Sample Server Response

```json
{
  "passwordStatus": 2,
  "passwordExpiration": "2025-12-31 00:00:00"
}
```

### Sample Server Failure Response

```json
{
  "error":{
    "code":222207916,
    "message":{
      "lang":"en-US",
      "value":"There is no User with that id: 123."
    }
  }
}
```

### Authentication

Basic Authentication using Login ID and Password

### Authorization

The user must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.
Reset the password on a user account

Purpose
Reset the password on a user account. Making this call changes the password and sets the passwordStatus to 2 (reset). The end user must change the password when he or she next logs in. Users can change their passwords either through the Web UI or through the User Details API.

URL
https://<myserver>:<port>/api/admin/users/{id}/resetpassword

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "newPassword": "temporary_password"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;newPassword&quot;</td>
<td>A temporary password provided by the administrator</td>
<td>Required</td>
<td>A string with a maximum length of 32 characters</td>
</tr>
</tbody>
</table>

Sample Request Payload

```json
{
    "newPassword": "tempsecret"
}
```

Sample Server Response

Status code: 204
Sample Server Failure Response

```
{
  "error":{
    "code":222207916,
    "message":{
      "lang":"en-US",
      "value":"There is no User with that id: 123."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.

Get permissions on a user account

Purpose

Returns permissions on a user account

URL

https://<myserver>:<port>/api/admin/users/{id}/permissions

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "roles": [integer, integer, ...],
    "permissions": [integer, integer, ...]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;roles&quot;</td>
<td>A role or list of roles associated with the user account. A user account must have at least one assigned role, and may only be assigned roles from its tenant.</td>
<td>The ID of the role assigned to the user account, or a comma separated list of role IDs assigned to the user account. See also Permissions and default roles on page 59.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A permission or list of permissions granted explicitly on the user account, in addition to those based on assigned roles.</td>
<td>The ID of the permission granted, or a comma-separated list of permissions granted, to the user account. See also Permissions and default roles on page 59.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

**Note:** The response will not return settings for optional properties that were not set in a previous POST or PUT request.

```
Status code: 200
Successful response

{
    "roles": [ 1, 2, ... ],
    "permissions": [ 1, 2, ... ]
}
```

Sample Server Failure Response

```
{
    "error":{
        "code":222207916,
        "message":{
            "lang":"en-US",
            "value":"There is no User with that id: 123."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password
Authorization
The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

Update permissions on a user account

Purpose
Updates permissions on a user account

URL
https://<myserver>:<port>/api/admin/users/{id}/permissions

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.

```
{
    "roles": [integer, integer, ...],
    "permissions": [integer, integer, ...]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;roles&quot;</td>
<td>A role or list of roles associated with the user account. A user account must have at least one assigned role, and may only be assigned roles from its tenant.</td>
<td>Required The ID of the role assigned to the user account, or a comma separated list of role IDs assigned to the user account. See also Permissions and default roles on page 59.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A permission or list of permissions granted explicitly on the user account, in addition to those based on assigned roles.</td>
<td>Optional The ID of the permission granted, or a comma-separated list of permissions granted, to the user account. See also Permissions and default roles on page 59.</td>
</tr>
</tbody>
</table>

Sample Request Payload

**Note:** Optional properties not included in the payload request will be removed from the object.

```
{
    "roles": [1, 2, ...],
    "permissions": [1, 2, ...]
}
```

Sample Server Success Response

```
Status code: 200
Successful response

{
    "roles": [1, 2, ...],
    "permissions": [1, 2, ...]
}
```

Sample Server Failure Response

```
{
    "error": {
        "code": 222207916,
        "message": {
            "lang": "en-US",
            "value": "There is no User with that id: 123."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.
Note: Administrator users cannot grant permissions they do not have to other user accounts.

Get authentication information

Purpose

Returns authentication information on a user account. The response includes the authentication user(s) and service(s) that belong to the user account.

URL

https://<myserver>:<port>/api/admin/users/{id}/authinfo

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "authUsers": [
        {
            "authUserName": "string",
            "authServiceId": integer
        },
        ...
    ]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;authUserName&quot;</td>
<td>The name of the authentication user.</td>
<td>Required</td>
<td>A string where the string specifies a user name persisted by the authentication service. The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;authServiceId&quot;</td>
<td>The ID of the authentication service against which the user is authenticating.</td>
<td>Required</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 is the ID for the default internal authentication service.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>x is an auto-generated ID for an external authentication service implemented by an administrator.</td>
</tr>
</tbody>
</table>

**Sample Server Response**

**Note:** The response will not return settings for optional properties that were not set in a previous POST or PUT request.

```
Status code: 200
Successful response

{
  "authUsers": [
    {
      "authUserName": "testuser",
      "authServiceId": 1
    }
  ]
}
```

**Sample Server Failure Response**

```
{
  "error":{
    "code":222207916,
    "message":{
      "lang":"en-US",
      "value":"There is no User with that id: 123."
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.
Update authentication information

Purpose
Updates authentication information on a user account. Allows an administrator to modify the authentication user(s) and service(s) that belong to the user account.

URL
https://<myserver>:<port>/api/admin/users/{id}/authinfo

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.

```json
{
   "authUsers": [
      {
         "authUserName": "string",
         "authServiceId": integer
      },
      ...
   ]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;authUserName&quot;</td>
<td>The name of the authentication user.</td>
<td>Required</td>
<td>A string where the string specifies a user name persisted by the authentication service. The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;authServiceId&quot;</td>
<td>The ID of the authentication service against which the user is authenticating.</td>
<td>Required</td>
<td>1</td>
</tr>
</tbody>
</table>

Sample Payload Request

```json
{
   "authUsers": [
   {
      "authUserName": "user_1",
      "authServiceId": 43
   },
   {
      "authUserName": "user_2",
      "authServiceId": 43
   }
]
}
```

Sample Server Response

Status code: 200
Successful response

```json
{
   "authUsers": [
   {
      "authUserName": "user_1",
      "authServiceId": 43
   },
   {
      "authUserName": "user_2",
      "authServiceId": 43
   }
]
}
```

Sample Server Failure Response

```json
{
   "error":{
      "code":222207916,
      "message":{
         "lang":"en-US",
         "value":"There is no User with that id: 123."
      }
   }
}
```
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the ModifyUsers (15) permission and administrative access on the tenant.

Get information on the authentication user

Purpose
Returns information on an authentication user

URL
https://<myserver>:<port>/api/admin/users/authUserName/{authUserName}
When the details query parameter is set to true, the response payload will include the tenantName property.
https://<myserver>:<port>/api/admin/users/authUserName/{authUserName}?details=true

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "authUserName" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;authUserName&quot;</td>
<td>The name of the authentication user.</td>
<td>A string where the string specifies a user name persisted by the authentication service. The maximum length is 128 characters.</td>
</tr>
</tbody>
</table>

Response Definition
The response takes the following format. The properties of the response are described in the table that follows.

```json
{
  "users": [
    {
      "id": user_account_id,
      "userName": "user_account_name",
      "tenantId": tenant_id,
      "tenantName": "tenant_name",
      "authUsername": "authentication_user_name",
```
"authServiceIds": [integer, integer, ...],
"statusInfo": {status_information},
"permissions": {permissions}
}

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;userName&quot;</td>
<td>The name of the user account.</td>
<td>The maximum length is 128 characters.</td>
</tr>
</tbody>
</table>
| "tenantId"   | The ID of the tenant to which the user belongs.                            | 1 | x  
1 | is the ID for the system tenant.                                               |
|              | x is the ID for a tenant created by an administrator. The ID is auto-generated when the tenant is created and cannot be changed. |
| "tenantName" | The name of the tenant to which the user belongs.                          | A string that specifies the name of the tenant.                              |
|              | Note: Included when the details query parameter is set to true (?details=true). |
| "authUserName" | The name of the authentication user                                       | A string where the string specifies a user name persisted by the authentication service. The maximum length is 128 characters. |
| "authServiceIds" | A list of authentication services which the authentication user can authenticate against | A comma separated list of authentication service IDs. See authenticationInfo Object on page 1157 for details. |
| "statusInfo" | The status of the user account defined by the status property and additional properties associated with an account lockout policy | See statusInfo Object on page 1155 for details. |
| "permissions" | Permissions associated with the user account in terms of the role(s) and permissions set explicitly on the account. User account permissions are the sum of the permissions on associated role(s) and permissions set explicitly on the account. | See permissions Object on page 1156 for details. |
Sample Server Success Response

**Note:** The response will not return settings for optional properties that were not set in a previous POST or PUT request.

```
Status code: 200
Successful response
```

```
{
   "users": [ 
   
   
   {
   "id": 3,
   "userName": "testuser",
   "tenantId": 1,
   "authUsername": "user_external",
   "authServiceIds": [2, 
   ],
   "statusInfo": {
   "status": 1,
   "accountLocked": false
   },
   "permissions": {
   "roles": [2]
   }
   }
   ]
}
```

Sample Server Failure Response

```
{
   "error": {
   "code": 222207916,
   "message": {
   "lang": "en-US",
   "value": "There is no User with that id: 123."
   }
   }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

**Get data sources for a user account**

**Purpose**

Retrieves a list of data sources for a user account.
**URL**
https://<myserver>:<port>/api/admin/users/{userid}/datasources

**Method**
GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Response Definition**

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "dataSources": [
        {
            "id": "datasource_id",
            "name": "datasource_name",
            "dataStore": "datastore_id",
            "isGroup": boolean,
            "description": "datasource_description",
            "sharedByAnotherUser": boolean,
            "sharedWithAnotherUser": boolean,
            "permissions": [integer, integer, ...]
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the data source</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the data source. This name is passed as a database parameter when establishing a connection to the data source with the ODBC driver, the JDBC driver, or the OData API.</td>
<td>The first character of the name must be a letter, and the name can contain only alphanumeric characters, underscores and dashes.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store on which the data source is being created. The data store defines the options that can be specified when creating the data source. Group data sources must be created on the Hybrid Data Pipeline group data store. A group data source is comprised of multiple member data sources that connect to one or more back end data stores such as Salesforce or SQL Server.</td>
<td>The integer ID of the data store Data store IDs can be obtained with the Get data stores call.</td>
</tr>
<tr>
<td>&quot;isGroup&quot;</td>
<td>Indicates whether the data source is a group data source. A group data source is comprised of member data sources.</td>
<td>true</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the data source</td>
<td>A description of the data source provided by the user who created the data source</td>
</tr>
<tr>
<td>&quot;sharedByAnotherUser&quot;</td>
<td>Indicates whether the data source is being shared by another user. Provided only when the data source is shared by another user.</td>
<td>true when the data source is being shared by another user.</td>
</tr>
<tr>
<td>&quot;sharedWithAnotherUser&quot;</td>
<td>Indicates whether the data source is being shared with another user. Provided only when the data source is shared with another user.</td>
<td>true when the data source is being shared with another user.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated explicitly with the data source. Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user. Any permissions specified for this data source will override the permissions for the user or the user's role that own this data source. You must specify the exact set of permissions that you want to set for this data source as no permissions are inherited from the user or user's role if permissions are specified on a data source. Permissions set on a group data source override permissions set on any of its member data sources.</td>
<td>A comma separated list of permission IDs See Data source permissions on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

Note: The response will not return settings for optional properties that were not set in a previous POST or PUT request.

Status code: 200
Successful response

```
{
  "dataSources": [
    {
      "id": 51,
      "name": "SF_test_ds_1",
      "dataStore": 1,
      "isGroup": false,
      "description": ""
    },
    ...
  ]
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222207004,
    "message": {
      "lang": "en-US",
      "value": "There is no DataSource with that id: 1234."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission or the ViewUsers (14) permission.

Get tenants administered

Purpose

Returns the list of tenants the user account administers

URL

https://<myserver>:<port>/api/admin/users/{id}/tenantsadministered

Method

GET
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "tenantsAdministered": [
        tenant_id,
        tenant_id,
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;tenantsAdministered&quot;</td>
<td>The ID or IDs of the tenants that the user administers.</td>
<td>A valid tenant ID or comma-separated list of valid tenant IDs.</td>
</tr>
</tbody>
</table>

Sample Server Response

Note: The response will not return settings for optional properties that were not set in a previous POST or PUT request.

Status code: 200
Successful response

```json
{
    "tenantsAdministered": [
        27,
        32
    ]
}
```

Sample Server Failure Response

```json
{  
    "error":{  
        "code":222207916,
        "message":{
            "lang":"en-US",
            "value":"There is no User with that id: 123."
        }
    }
}
Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the ViewUsers (14) permission and administrative access on the tenant.

Update tenants administered

Purpose
Updates the list of tenants the account administers

URL
https://<myserver>:<port>/api/admin/users/{id}/tenantsadministered

Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter "id" described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "tenantsAdministered": [
        tenant_id,
        tenant_id,
        tenant_id,
        ...
    ]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;tenantsAdministered&quot;</td>
<td>The ID or IDs of the tenants that the user administers.</td>
<td>A valid tenant ID or comma-separated list of valid tenant IDs.</td>
</tr>
</tbody>
</table>

**Sample Payload Request**

```json
{
    "tenantsAdministered": [
        27,
        32,
        37
    ]
}
```

**Sample Server Response**

```
Status code: 200
Successful response
```

```
{
    "tenantsAdministered": [
        27,
        32,
        37
    ]
}
```

**Sample Server Failure Response**

```
{
    "error":{
        "code":222207916,
        "message":{
            "lang":"en-US",
            "value":"There is no User with that id: 123."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission; or the user must have the TenantAPI (25) permission, ModifyUsers (15) permission, and administrative access on the tenant.
User Details API

The User Details API allows users to change their passwords. A user must have either the Administrator (12) permission or the ChangePassword (9) permission to change his or her password. By default, users must provide a current password as well as a new password when changing passwords. The following table summarizes the operation.

Note: Hybrid Data Pipeline also supports change password functionality where the user is not required to enter a current password. Administrators can enable this non-default behavior with the System Configurations API on page 1124.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updates a user password</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/userdetails/changePassword</td>
</tr>
</tbody>
</table>

Change password

Purpose
Updates the password on the user account

URL
https://<myserver>:<port>/api/admin/userdetails/changePassword

Method
PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "currentPassword": "current_password",
    "newPassword": "new_password"
}
```
### Parameter Description Usage Valid Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;currentPassword&quot;</td>
<td>Specifies the current password.</td>
<td>Required</td>
<td>The password length can be from 1 to 32 characters.</td>
</tr>
<tr>
<td>&quot;newPassword&quot;</td>
<td>Specifies a new password.</td>
<td>Required</td>
<td>The password length can be from 1 to 32 characters.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```json
{
   "currentPassword": "Secret",
   "newPassword": "NewSecret"
}
```

**Note:** Hybrid Data Pipeline also supports change password functionality where the user is not required to enter a current password when changing passwords. Administrators can enable this non-default behavior with the System Configurations API. If the `secureChangePassword` attribute is set to false, the request payload for change password functionality should only include "newPassword": "<mynewpassword>".

**Sample Server Response**

```json
{
   "passwordStatus": 2,
   "passwordExpiration": "2025-12-31 00:00:00"
}
```

**Sample Server Failure Response**

```json
{
   "error":{
      "code":222207916,
      "message":{
         "lang":"en-US",
         "value":"There is no User with that id: 123."
      }
   }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission or the ChangePassword (9) permission.

**Health Check API**

The Health Check API can be used to configure a load balancer to perform periodic health checks on nodes in a Hybrid Data Pipeline cluster (see also Load balancer configuration on page 38).
### Get health check

**Purpose**
Performs a health check on the node or nodes running the data access service. Permits the configuration of a load balancer to perform periodic health checks on cluster nodes (see also Load balancer configuration on page 38). If the service is running as expected, the status code 200 and the status message `active` are returned. Other responses should be investigated.

**URL**
https://<myserver>:<port>/api/healthcheck

**Method**
GET

**URL Parameters**
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Definition**
The response takes the following format. The parameters of the response are described in the table that follows.

```
{
    "status": "<status>"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;status&quot;</td>
<td>The status of the specified node</td>
<td>The value <code>active</code> indicates that the node is running as expected. Other responses should be investigated.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

Status code: 200
Successful response

{
  "status": active
}

Sample Server Failure Response

If no response is returned, the operation will time out.

Failed connect to 172.29.37.229:8443; Connection timed out.

Authentication
This endpoint is accessible to any user. Does not require authentication.

Authorization
Any active Hybrid Data Pipeline user.

Head health check

Purpose
Performs a health check on the node or nodes running the data access service. Permits the configuration of a load balancer to perform periodic health checks on cluster nodes (see also Load balancer configuration on page 38). If the service is running as expected, the status code 200 is returned. Other responses should be investigated.

URL
https://<myserver>:<port>/api/healthcheck

Method
HEAD

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition
The status code 200 with an empty response body is returned.
Sample Server Success Response

Status code: 200
Successful response

Sample Server Failure Response
If no response is returned, the operation will time out.
Failed connect to 172.29.37.229:8443; Connection timed out.

Authentication
This endpoint is accessible to any user. Does not require authentication.

Authorization
Any active Hybrid Data Pipeline user.

IP Address Whitelist API
The IP Address Whitelist API allows users to create an IP address whitelist to determine which IP addresses (either individual IP addresses or a range of IP addresses) can access resources such as the Management API, the Administrators API, data access, and the Web UI. Depending on a user's permissions, IP address whitelists can be implemented at system, tenant, and user levels.

• A user with the Administrator (12) permission (a system administrator) can implement and create whitelists for all resources at system, tenant and user levels.

• A user with the following permissions can create whitelists for resources at the tenant level: the MgmtAPI (11) permission, the IPWhiteList (29) permission, and administrative access to the tenant.

• A user with the following permissions can create whitelists for resources at the user level: the Mgmt (11) permission and the IPWhiteList (29) permission.

IP address whitelisting functionality is enabled by default. Administrators can change this behavior using attribute 8 of the System Configurations API. For details, see Disabling and enabling the IP whitelisting feature.

Note: If you accidently block your IP address while setting IP whitelists, you need to invoke IP from the Hybrid Data Pipeline server machine. To bypass the whitelist filter, you must use localhost or 127.0.0.1 or ::1 in the URL.

You can perform the following operations with the IP Address Whitelist API.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve IP address whitelists at the system level on page 1196</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/system</td>
</tr>
<tr>
<td>Update IP address whitelists at the system level on page 1202</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/system</td>
</tr>
<tr>
<td>Operation</td>
<td>Request</td>
<td>URL</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Create IP address whitelists at the system level on page 1199</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/system</td>
</tr>
<tr>
<td>Delete IP address whitelists at the system level on page 1205</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/system</td>
</tr>
<tr>
<td>Retrieve tenants configured with IP address whitelists on page 1205</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/tenants</td>
</tr>
<tr>
<td>Retrieve IP address whitelists for a tenant on page 1208</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/tenants/{id}</td>
</tr>
<tr>
<td>Update IP address whitelists for a tenant on page 1213</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/tenants/{id}</td>
</tr>
<tr>
<td>Create IP address whitelists for a tenant on page 1210</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/tenants/{id}</td>
</tr>
<tr>
<td>Delete IP address whitelists for a tenant on page 1216</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/tenants/{id}</td>
</tr>
<tr>
<td>Retrieve users configured with whitelist IP on page 1217</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/admin/security/whitelist/users</td>
</tr>
<tr>
<td>Retrieve IP address whitelists for the authenticated user on page 1219</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/security/whitelist/user</td>
</tr>
<tr>
<td>Update IP address whitelists for a user on page 1223</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/security/whitelist/user</td>
</tr>
<tr>
<td>Create IP address whitelists for the authenticated user on page 1221</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/security/whitelist/user</td>
</tr>
<tr>
<td>Delete IP address whitelists for the authenticated user on page 1226</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/security/whitelist/user</td>
</tr>
</tbody>
</table>

Retrieve IP address whitelists at the system level

Purpose
Returns IP address whitelists for resources which are configured at the system level.
**URL**
https://<myserver>:<port>/api/admin/security/whitelist/system

**Method**
GET

**URL Parameters**
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Payload Definition**
The response takes the following format. The properties of the response are described in the table that follows.

```
{
    "managementAPI": [
        {
            "startAddress": "<start_ip_address>",
            "endAddress": "<end_ip_address>"
        }
    ],
    "adminAPI": [...],
    "dataAccess": [...],
    "webUI": [...]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;managementAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Management API.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;adminAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;dataAccess&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;webUI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Web UI.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
</tbody>
</table>

**Note:** Can only be applied at the system level.

---

**Sample Server Success Response**

Status code: 200
Successful response

```json
{
   "managementAPI": [ 
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.30.10"
      }
   ],
   "adminAPI": [ 
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.40.10"
      }
   ],
   "dataAccess": [ 
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.50.10"
      }
   ],
   "webUI": [ 
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.50.10"
      }
   ]
}```
Sample Server Failure Response

```json
{
  "error": {
    "code": 222208712,
    "message": "Problem getting WhiteList IPs at this time. Please try again at another time."
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

**Create IP address whitelists at the system level**

**Purpose**

Sets IP address whitelists for different resources at a system level.

**URL**

https://<myserver>:<port>/api/admin/security/whitelist/system

**Method**

POST

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Request Payload Definition**

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
  "managementAPI": [
    {
      "startAddress": "<start_ip_address>",
      "endAddress": "<end_ip_address>"
    },
    "adminAPI": [...],
    "dataAccess": [...]
  ]
}
```
ValidValuesUsageDescriptionProperty

An array of JSON objects. Each object must be either a single IP address designated with the "startAddress" property, or a range of IP addresses designated with the "startAddress" and "endAddress" properties. IP addresses may be specified in either IPv4 or IPv6.

Optional

Individual IP addresses or a range of IP addresses that restrict access to the Management API.

"managementAPI"

Optional

Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.

"adminAPI"

Optional

Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.

"dataAccess"

Optional

Individual IP addresses or a range of IP addresses that restrict access to the Web UI.

"webUI"

Note: Can only be applied at the system level.

Request Payload Sample

```json
{
    "managementAPI": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.30.10"
        }
    ],
    "adminAPI": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.40.10"
        }
    ],
    "dataAccess": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.50.10"
        }
    ]
}
```
Sample Server Success Response

Status code: 201
Successful response

{  "managementAPI": [
   {  
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.30.10"
   }
  ],  
  "adminAPI": [
   {  
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.40.10"
   }
  ],  
  "dataAccess": [
   {  
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.50.10"
   }
  ],  
  "webUI": [
   {  
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.50.10"
   }
  ]
}

Sample Server Failure Response

{  "error":  
   {  
      "code": 222208711,
      "message": "Problem creating WhiteList IPs at this time. Please try again at another time."
   }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.
Update IP address whitelists at the system level

**Purpose**
Updates IP address whitelists at the system level.

**URL**
https://<myserver>:<port>/api/admin/security/whitelist/system

**Method**
PUT

**URL Parameters**
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Request Payload Definition**
The request takes the following format. The properties of the request are described in the table that follows.

```json
{
  "managementAPI": [
    {
      "startAddress": "<start_ip_address>",
      "endAddress": "<end_ip_address>"
    }
  ],
  "adminAPI": [...],
  "dataAccess": [...],
  "webUI": [...]}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;managementAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Management API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;adminAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;dataAccess&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;webUI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Web UI.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
</tbody>
</table>

**Note:** Can only be applied at the system level.

**Request Payload Sample**

**Note:** Optional properties not included in the payload request will be removed from the object.

```json
{
   "managementAPI": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.30.10"
      }
   ],
   "adminAPI": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.40.10"
      }
   ],
   "dataAccess": [
      {
         "startAddress": "10.20.30.0",
         "endAddress": "10.20.50.10"
      }
   ]
}
```
Sample Server Success Response

Status code: 200
Successful response

Sample Server Failure Response

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.
Delete IP address whitelists at the system level

**Purpose**
Deletes IP address whitelists configured at the system level.

**URL**
https://<myserver>:<port>/api/admin/security/whitelist/system

**Method**
DELETE

**URL Parameters**
&lt;myserver&gt; is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, &lt;port&gt; is the port number specified as the Server Access Port during installation. For a load balancer installation, &lt;port&gt; must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Sample Server Success Response**

Status code: 204
Successful response

**Sample Server Failure Response**

```json
{
    "error": {
        "code": 222208715,
        "message": {
            "lang": "en-US",
            "value": "Problem deleting WhiteList IPs at this time. Please try again at another time."
        }
    }
}
```

**Authentication**
Basic Authentication using Login ID and Password

**Authorization**
The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Retrieve tenants configured with IP address whitelists

**Purpose**
Retrieves tenants that are configured with IP address whitelists.
Note: The response returns the tenants that are accessible by the user making the request. If a system administrator (user with Administrator permission) makes the request, the response lists all the tenants in the system that have IP address whitelists. If a tenant administrator makes the request, the response lists only the tenants (with IP address whitelists) for which the tenant administrator has administrative access.

URL

https://<myserver>:<port>/api/admin/security/whitelist/tenants

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Payload Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
  "appliedWhiteLists": [
  {
    "id": "tenant_id",
    "name": "tenant_name",
    "protectedResources": [
      "resource_name",
      "resource_name",
      ...
    ],
  },
  ...
]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the tenant</td>
<td>A string that specifies the name of the tenant.</td>
</tr>
<tr>
<td>&quot;protectedResources&quot;</td>
<td>A list of protected resources.</td>
<td>One more or more valid protected resources. Protected resources include the managementAPI, adminAPI, dataAccess, or webUI.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

If a system administrator (user with Administrator permission) makes the request, the response lists all the tenants in the system that have IP address whitelists.

Status code: 200
Successful response

```json
{
  "appliedWhiteLists": [
    {
      "id": 1,
      "name": "Tenant1",
      "protectedResources": [
        "managementAPI",
        "dataAccess"
      ]
    },
    {
      "id": 2,
      "name": "Tenant2",
      "protectedResources": [
        "managementAPI"
      ]
    }
  ]
}
```

If a tenant administrator makes the request, the response lists only the tenants (with IP address whitelists) for which the tenant administrator has administrative access.

Status code: 200
Successful response

```json
{
  "appliedWhiteLists": [
    {
      "id": 48,
      "name": "OrgH",
      "protectedResources": [
        "managementAPI",
        "dataAccess"
      ]
    }
  ]
}
```

Sample Server Failure Response

```json
{
  "error": {
    "code": 222208712,
    "message": {
      "lang": "en-US",
      "value": "Problem getting WhiteList IPs at this time. Please try again at another time."
    }
  }
}
```

Authentication

Basic Authentication using Login ID and Password
Authorization
The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Retrieve IP address whitelists for a tenant

Purpose
Retrieves IP address whitelists for a tenant.

URL
https://<myserver>:<port>/api/admin/security/whitelist/tenants/{id}

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

Response Payload Definition
The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "managementAPI": [
        {
            "startAddress": "<start_ip_address>",
            "endAddress": "<end_ip_address>"
        }
    ],
    "adminAPI": [...],
    "dataAccess": [...],
    "webUI": [...]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;managementAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Management API.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;adminAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;dataAccess&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;webUI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Web UI.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6. Note: Can only be applied at the system level.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

**Status code**: 200  
**Successful response**

```json
{
    "managementAPI": [ 
        { "startAddress": "10.20.30.0"
    },
    "adminAPI": [],
    "dataAccess": [ 
        { "startAddress": "10.20.30.0",
          "endAddress": "10.20.30.20"
        }
    ],
    "webUI": null
}
```
Sample Server Success Response

Status code: 200
Successful response

{
    "managementAPI": [
        {
            "startAddress": "10.20.30.0"
        }
    ],
    "adminAPI": [],
    "dataAccess": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.30.20"
        }
    ],
    "webUI": null
}

Sample Server Failure Response

{
    "error": {
        "code": 222208720,
        "message": {
            "lang": "en-US",
            "value": "tenant id : {0} does not exist."
        }
    }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission; or the user must have the MgmtAPI (11) permission, the IPWhiteList (29) permission, and administrative access for the tenant.

Create IP address whitelists for a tenant

Purpose

Creates IP address whitelists for a tenant.

URL

https://<myserver>:<port>/api/admin/security/whitelist/tenants/{id}

Method

POST
**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

**Request Payload Definition**

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
   "managementAPI": [
      {
         "startAddress": "<start_ip_address>",
         "endAddress": "<end_ip_address>"
      }
   ],
   "adminAPI": [...],
   "dataAccess": [...],
   "webUI": [...]
}
```
Valid Values Usage Description Property

Valid Values Usage Description Property

**managementAPI**
- An array of JSON objects. Each object must be either a single IP address designated with the "startAddress" property, or a range of IP addresses designated with the "startAddress" and "endAddress" properties. IP addresses may be specified in either IPv4 or IPv6.
- Individual IP addresses or a range of IP addresses that restrict access to the Management API.

**adminAPI**
- An array of JSON objects. Each object must be either a single IP address designated with the "startAddress" property, or a range of IP addresses designated with the "startAddress" and "endAddress" properties. IP addresses may be specified in either IPv4 or IPv6.
- Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.

**dataAccess**
- An array of JSON objects. Each object must be either a single IP address designated with the "startAddress" property, or a range of IP addresses designated with the "startAddress" and "endAddress" properties. IP addresses may be specified in either IPv4 or IPv6.
- Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.

**webUI**
- An array of JSON objects. Each object must be either a single IP address designated with the "startAddress" property, or a range of IP addresses designated with the "startAddress" and "endAddress" properties. IP addresses may be specified in either IPv4 or IPv6.
- Individual IP addresses or a range of IP addresses that restrict access to the Web UI.
- Note: Can only be applied at the system level.

**Request Payload Sample**

```json
{
    "managementAPI": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.30.10"
        }
    ],
    "adminAPI": [],
    "dataAccess": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.50.10"
        }
    ],
    "webUI": null
}
```
Sample Server Success Response

Status code: 201
Successful response

{
    "managementAPI": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.30.10"
        }
    ],
    "adminAPI": [],
    "dataAccess": [
        {
            "startAddress": "10.20.50.0",
            "endAddress": "10.20.50.10"
        }
    ],
    "webUI": null
}

Sample Server Failure Response

{
    "error": {
        "code": 222208718,
        "message": {
            "lang": "en-US",
            "value": "WhiteList IPs already exists for tenant id: {0}."}
    }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Update IP address whitelists for a tenant

Purpose
Updates IP address whitelists for a tenant.

URL
https://<myserver>:<port>/api/admin/security/whitelist/tenants/{id}

Method
PUT
URL Parameters

\(<myserver>\) is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, \(\langle port\rangle\) is the port number specified as the Server Access Port during installation. For a load balancer installation, \(\langle port\rangle\) must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter \{id\} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "managementAPI": [
        {
            "startAddress": "<start_ip_address>",
            "endAddress": "<end_ip_address>"
        }
    ],
    "adminAPI": [...],
    "dataAccess": [...],
    "webUI": [...]
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;managementAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Management API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;adminAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;dataAccess&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;webUI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Web UI.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
</tbody>
</table>

**Note:** Can only be applied at the system level.

---

**Request Payload Sample**

**Note:** Optional properties not included in the payload request will be removed from the object.

```json
{
    "managementAPI": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.30.10"
        }
    ],
    "adminAPI": [],
    "dataAccess": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.50.10"
        }
    ],
    "webUI": null
}
```
Sample Server Success Response

Status code: 200
Successful response

{
  "managementAPI": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.30.10"
    }
  ],
  "adminAPI": [],
  "dataAccess": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.50.10"
    }
  ],
  "webUI": null
}

Sample Server Failure Response

{
  "error": {
    "code": 222208719,
    "message": {
      "lang": "en-US",
      "value": "whiteList IPs does not exist for tenant id : {0}."
    }
  }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Delete IP address whitelists for a tenant

Purpose
Deletes IP address whitelists for a tenant.

URL
https://<myserver>;<port>/api/admin/security/whitelist/tenants/{id}

Method
DELETE
### URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described in the following table is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the tenant.</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>

### Sample Server Success Response

Status code: 204
Successfully deleted the whitelist IPs for the given tenant id

### Sample Server Failure Response

```json
{
   "error": {
      "code": 222208720,
      "message": {
         "lang": "en-US",
         "value": "tenant id : {0} does not exist."
      }
   }
}
```

### Authentication

Basic Authentication using Login ID and Password

### Authorization

The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

### Retrieve users configured with whitelist IP

#### Purpose

Retrieves all users that are configured with IP address whitelists.

**Note:** The response returns the users that the administrator making the request can administer. If a system administrator (user with Administrator permission) makes the request, the response lists all the users in the system that have IP address whitelists. If a tenant administrator makes the request, the response lists only the users (with IP address whitelists) in tenants for which tenant administrator has administrative access.

#### URL

https://<myserver>:<port>/api/admin/security/whitelist/users

IP address whitelist details for a specific user can be retrieved by appending the URL with the ?user query parameter and specifying a user name. For example:
https://<myserver>:<port>/api/admin/security/whitelist/users?user=TestUserA

Method
GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Payload Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
  "appliedWhiteLists": [
    {
      "id": user_id,
      "name": "user_name",
      "protectedResources": [
        "resource_name",
        "resource_name",
        ...
      ],
      ...
    }
  ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the user account.</td>
<td>The maximum length is 128 characters.</td>
</tr>
<tr>
<td>&quot;protectedResources&quot;</td>
<td>A list of protected resources.</td>
<td>One more or more valid protected resources. Protected resources include the managementAPI, adminAPI, dataAccess, or webUI.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```json
{
  "appliedWhiteLists": [
    {
      "id": 66,
      "name": "User303",
      "protectedResources": [
        "managementAPI",
        "dataAccess"
      ]
    },
    ...
  ]
}
```
[ 
  "id": 124,
  "name": "User606",
  "protectedResources": [
    "managementAPI"
  ]
]

Sample Server Failure Response

{ 
  "error": {
    "code": 222208712,
    "message": {
      "lang": "en-US",
      "value": "Problem getting WhiteList IPs at this time. Please try again at another time."
    }
  }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Retrieve IP address whitelists for the authenticated user

Purpose

Returns IP address whitelists for the authenticated user.

URL

https://<myserver>:<port>/api/mgmt/security/whitelist/user

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
**Response Payload Definition**

The response takes the following format. The properties of the response are described in the table that follows.

```
{
    "managementAPI": [
        {
            "startAddress": "<start_ip_address>",
            "endAddress": "<end_ip_address>"
        }
    ],
    "adminAPI": [],
    "dataAccess": [],
    "webUI": []
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;managementAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Management API.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;adminAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;dataAccess&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;webUI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Web UI.</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
</tbody>
</table>

**Note:** Can only be applied at the system level.

---

**Sample Server Success Response**

Status code: 200
Successful response

```
{
    "managementAPI": [
        {
            "startAddress": "10.20.30.0"
        }
    ],
    "adminAPI": []
}
```
Sample Server Failure Response

```json
{
    "error": {
        "code": 222207916,
        "message": {
            "lang": "en-US",
            "value": "There is no User with that id: 34."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Create IP address whitelists for the authenticated user

Purpose

Creates IP address whitelists for the authenticated user.

URL

```
https://<myserver>:<port>/api/mgmt/security/whitelist/user
```

Method

POST

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "managementAPI": [
        {
            "dataAccess": [
                {
                    "startAddress": "10.20.30.0",
                    "endAddress": "10.20.30.20"
                }
            ],
            "webUI": null
        }
    }
}
```
### ValidValuesUsageDescriptionProperty

An array of JSON objects. Each object must be either a single IP address designated with the "startAddress" property, or a range of IP addresses designated with the "startAddress" and "endAddress" properties. IP addresses may be specified in either IPv4 or IPv6.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;managementAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Management API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;adminAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;dataAccess&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;webUI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Web UI.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
</tbody>
</table>

**Note:** Can only be applied at the system level.

### Request Payload Sample

```json
{
  "managementAPI": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.30.10"
    }
  ],
  "adminAPI": [],
  "dataAccess": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.50.10"
    }
  ]
}
```

---

Chapter 10: Hybrid Data Pipeline API reference

Sample Server Success Response

Status code: 201
Successful response

{
  "managementAPI": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.30.10"
    }
  ],
  "adminAPI": [],
  "dataAccess": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.50.10"
    }
  ],
  "webUI": null
}

Sample Server Failure Response

{
  "error": {
    "code": 222208711,
    "message": {
      "lang": "en-US",
      "value": "Problem creating WhiteList IPs at this time. Please try again at another time."
    }
  }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Update IP address whitelists for a user

Purpose
Updates IP address whitelists for the authenticated user.

URL
https://<myserver>:<port>/api/mgmt/security/whitelist/user
Method

PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "managementAPI": [
        {
            "startAddress": "<start_ip_address>",
            "endAddress": "<end_ip_address>"
        }
    ],
    "adminAPI": [...],
    "dataAccess": [...],
    "webUI": [...]  
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;managementAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Management API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;adminAPI&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict access to the Administrators API.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
<tr>
<td>&quot;dataAccess&quot;</td>
<td>Individual IP addresses or a range of IP addresses that restrict data access through JDBC, ODBC, and OData calls.</td>
<td>Optional</td>
<td>An array of JSON objects. Each object must be either a single IP address designated with the &quot;startAddress&quot; property, or a range of IP addresses designated with the &quot;startAddress&quot; and &quot;endAddress&quot; properties. IP addresses may be specified in either IPv4 or IPv6.</td>
</tr>
</tbody>
</table>
| "webUI"       | Individual IP addresses or a range of IP addresses that restrict access to the Web UI. | Optional         | An array of JSON objects. Each object must be either a single IP address designated with the "startAddress" property, or a range of IP addresses designated with the "startAddress" and "endAddress" properties. IP addresses may be specified in either IPv4 or IPv6. 

**Note:** Can only be applied at the system level.

---

**Request Payload Sample**

**Note:** Optional properties not included in the payload request will be removed from the object.

```json
{
    "managementAPI": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.30.10"
        }
    ],
    "adminAPI": [],
    "dataAccess": [
        {
            "startAddress": "10.20.30.0",
            "endAddress": "10.20.50.10"
        }
    ],
    "webUI": null
}
```
Sample Server Success Response

Status code: 200
Successful response

{
  "managementAPI": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.30.10"
    }
  ],
  "adminAPI": [],
  "dataAccess": [
    {
      "startAddress": "10.20.30.0",
      "endAddress": "10.20.50.10"
    }
  ],
  "webUI": null
}

Sample Server Failure Response

{
  "error": {
    "code": 222208713,
    "message": {
      "lang": "en-US",
      "value": "Problem updating WhiteList IPs at this time. Please try again at another time."
    }
  }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Delete IP address whitelists for the authenticated user

Purpose
Deletes IP address whitelists for the authenticated user.

URL
https://<myserver>:<port>/api/mgmt/security/whitelist/user

Method
DELETE
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Sample Server Success Response

Status code: 204
Successful response

Sample Server Failure Response

{
  "error": {
    "code": 222208715,
    "message": {
      "lang": "en-US",
      "value": "Problem deleting WhiteList IPs at this time. Please try again at another time."}
  }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

The user must have the Administrator (12) permission, or the MgmtAPI (11) and IPWhiteList (29) permissions.

Management API

The Management API gives administrators and users the ability to create and manage Hybrid Data Pipeline data sources, manage On-Premises Connector access control lists (ACLs), manage OAuth Tokens for Google Analytics data sources, and integrate OAuth 2.0 with client applications.

Connector API

The Connector API can be used to manage access to backend data through On-Premises Connectors. An On-Premises Connector is, by default, only accessible to its owner, the user who installed and registered the connector. The owner may share backend data by authorizing other Hybrid Data Pipeline users to use an On-Premises Connector. The owner manages a list of users that can use the connector by executing GET, POST, PUT, and DELETE operations with the Connector API.

In addition, the Connector API allows users to manage requests among multiple On-Premises Connectors by creating On-Premises Connector groups. See the following topics for details.

• Using Failover and Balancing Requests with an On-Premises Connector Group on page 1229
• Configuring Failover and Balancing Requests with an On-Premises Connector Group on page 1230
The following operations can be performed with the Connector API.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve a list of On-Premises Connectors owned by or shared with the authenticated user</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors</td>
</tr>
<tr>
<td>Retrieve the On-Premises Connector's information</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;</td>
</tr>
<tr>
<td>Update the connector information</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;</td>
</tr>
<tr>
<td>Retrieve authorized users for a particular connector</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/authuser</td>
</tr>
<tr>
<td>Add authorized users to a connector’s access control list.</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/authuser</td>
</tr>
<tr>
<td>Update the list of authorized users for a connector</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/authuser</td>
</tr>
<tr>
<td>Delete authorized users and groups from a connector’s access control list</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/authuser</td>
</tr>
<tr>
<td>Create a Connector group to enable failover to member Connectors</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors</td>
</tr>
<tr>
<td>Add members to a Connector group</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/members</td>
</tr>
<tr>
<td>Retrieve the list of members for a Connector group</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/members</td>
</tr>
<tr>
<td>Define how round-robin load balancing is implemented in a Group Connector</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;group-connector&gt;</td>
</tr>
<tr>
<td>Replace the list of On-Premises Connectors in a Connector group</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/members</td>
</tr>
<tr>
<td>Operation</td>
<td>Request</td>
<td>URL</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>-----</td>
</tr>
<tr>
<td>Remove an On-Premises Connector from an On-Premise Connector Group</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;/members</td>
</tr>
<tr>
<td>Deprecated</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/connectors/&lt;connector-ID&gt;</td>
</tr>
</tbody>
</table>

Using Failover and Balancing Requests with an On-Premises Connector Group

Users can define groups of On-Premises Connectors using the Connector API. When defining an On-Premises data source, a Connector Group ID can be specified as the Connector ID of the data source instead of specifying a Connector ID for a single connector. This allows you to use failover and balance requests among multiple On-Premises Connectors. For configuration details, see Configuring Failover and Balancing Requests with an On-Premises Connector Group on page 1230.

Connection Failover

Connection time failover is supported for queries executed from the Hybrid Data Pipeline ODBC and JDBC drivers, and from the OData API, because users can now define groups of On-Premise Connectors. When defining an on-premise data source, a Connector Group ID can be specified as the Connector ID of the data source instead of specifying a Connector ID for a single connector. When a Group Connector ID is used for a data source and a connection is requested for that data source the connectivity service uses the first On-Premises Connector in the group to establish the connection.

If the connection succeeds, the application is successfully connected to the data source. If the connection fails, the connectivity service then uses the next On-Premises Connector in the group to attempt to connect to the data source. If that connection fails, the next connector in the group is used. This continues until a connection is successfully established or all of the On-Premises Connectors in the group have been tried and failed. In the latter case, an error is returned to the application.

Failover is also supported at execute time for a fetch operation. If a SELECT statement is executed using either driver, and a connection failure or connection timeout occurs during the execution of the statement, the driver triggers the failover sequence in an attempt to reconnect to the data source. If the connection is re-established, the SELECT statement is re-executed. Select failover is not supported if a query is executed through the OData API.

Balancing Requests

An On-Premises Connector group can be configured to balance requests across multiple connectors. By default, a connector group enabled to balance requests attempts to balance requests equally across the connectors in the connector group. Optionally, a weight can be assigned to one or more members of the group. This allows more traffic to be directed to a specific connector if needed. For example, if Connector1 is running on a faster server than Connector2, a higher number of requests can be sent to Connector1. A round-robin algorithm is used to support this method for balancing requests.
Configuring Failover and Balancing Requests with an On-Premises Connector Group

To enable failover and balance requests with an On-Premises Connector Group, you must first have multiple On-Premises Connectors installed. (See the Progress DataDirect Hybrid Data Pipeline Installation Guide.) Once multiple On-Premises Connectors have been installed, take the following steps to enable failover and balance requests.

1. Create an On-Premises Connector Group by executing a POST request to the `<base>/connectors` endpoint. (See Create a Connector Group on page 1249 for further details, including code samples and parameter descriptions.)
   a) To configure failover, set the `connectionTimeout` and `retryDelay` properties in the request to the desired values.
   b) To enable round-robin request balancing, set the `loadBalancing` property in the request to `Round Robin`.
   c) To configure round-robin request balancing, specify the `weight` for each member of the Connector Group in the `members` array. (Setting `weight` is optional. The default value for `weight` is 1.)

2. Note the Connector ID for the group that is returned in the POST response. This is the Connector Group ID.

3. Create a new data source or modify an existing one to use the Connector Group. (See Create a data source on page 1300 and Update a data source on page 1315 for details.)
   a) Set the Connector ID for the data source to the Connector Group ID. The Connector Group ID is the Connector ID returned in the POST request that created the group.

Get Connectors

**Purpose**

Retrieves the list of On-Premises Connectors that are owned by or shared with the authenticated Hybrid Data Pipeline user.

**URL**

```
https://<myserver>:<port>/api/mgmt/connectors
```

where `<myserver>` is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed, and `<connector-ID>` is a unique value associated with the On-Premises Connector.

---

**Note:** Unless the ports 80 and 443 are redirected to 8080 and 8443 respectively, you must specify `<myserver>:<port>`.

**Method**

GET
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Query Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;details&quot;</td>
<td>Determines whether the additional details of the Connectors are included in the response.</td>
<td>Optional</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>When set to true, the details of Connectors, such as the owner and authorized users, are included in the response.</td>
<td></td>
<td>When set to false, only the Connector IDs are returned.</td>
</tr>
<tr>
<td></td>
<td>When set to false, only the Connector IDs are returned.</td>
<td></td>
<td>The default is false.</td>
</tr>
<tr>
<td>&quot;accessible&quot;</td>
<td>Determines whether the response includes all of the Connectors owned by or shared with the user making the request.</td>
<td>Optional</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>When set to true, the response includes all the Connectors accessible (owned and shared) to the user making this request.</td>
<td></td>
<td>When set to false, the response includes only the Connectors owned by this user.</td>
</tr>
<tr>
<td></td>
<td>The default is false.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sample URL

https://<myserver>:<port>/api/mgmt/connectors?details=true&accessible=true

Response Definition

When the query parameters are set to false (the default), only the Connector ID of each Connector owned by the user is returned. The response has the following format:

```json
{
    "connectorIDs": [
        "connector-id",
        "connector-id",
        ...
        "connector-id"
    ]
}
```
where

`connector-id`

can reference an individual On-Premises Connector or an On-Premises Connector Group owned by or shared with the authenticated user. If the authenticated user does not own any Connectors and no Connectors have been shared, the array of connector IDs in the response is empty.

When both query parameters are set to true, the Connector IDs and details of each Connector owned by and shared with the user are returned. The response has the following format:

```json
{
  "connectorIDs": [
    {
      "connectorId": "<connector-id>",
      "owner": "ownername1",
      "label": "Label1",
      "authUser": ["user1", "user2"]
    },
    {
      "connectorId": "<connector-id>",
      "owner": "ownername1",
      "label": "Label2",
      "authUser": ["user1", "user2"]
    },
    ...
    {
      "connectorId": "<connector-id>",
      "owner": "ownername1",
      "label": "Label3",
      "authUser": ["user3", "user4"]
    }
  ]
}
```
Sample Server Response

In this example, details and accessible were set to true. The first Connector lists three authorized users, indicating that the user is the owner of the Connector. Note that the second Connector has a different owner, and therefore, has an empty list of authorized users.

```json
{
    "connectorIds": [
        {
            "connectorId": "55b55556-22d1-4f6a-888f-444a2df565e0",
            "owner": "ddctest01",
            "label": "DevTest1",
            "authUser": [
                "Joe",
                "Fred",
                "Tom"
            ]
        },
        {
            "connectorId": "7e7afa77-5555-44c3-b0ff-6b888edf8a",
            "owner": "ddctest02",
            "label": "DevTest1",
            "authUser": []
        }
    ]
}
```

Also notice that because the Connectors have the same label, the owner's name will be attached to the label of the second Connector when displayed in the data source setup screen.

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

Any active Hybrid Data Pipeline user

**See also**

Get Connector Information on page 1233

**Get Connector Information**

**Purpose**

Allows the owner to retrieve information about the members of one or more On-Premises Connectors in a connector group: member ID, sequence, and weight.

**URL**

https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>

where

<myserver> is the DSN name or the IP address of the machine where Hybrid Data Pipeline is installed.

**Note:** Unless the ports 80 and 443 are redirected to 8080 and 8443 respectively, you must specify <myserver>:<port>.
<connector-ID> is a unique value associated with the On-Premises Connector. The value is returned using the https://<myserver>:<port>/api/mgmt/connectors GET request. The authorized user must be the owner of the On-Premises Connector specified.

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Definition**

The response has the following format:

```
{
   "connectorID": "connector-id",
   "owner": "owner-user-id",
   "label": "Label",
   "connectorGroup": {
      "connectionTimeout": 15,
      "retryDelay": 120,
      "loadBalancing": "Round Robin",
      "members": [
         {
            "memberID": "memberId",
            "sequence": 1,
            "weight": 1
         }
      ],
   }
   "authUser": [
      "authorized-user",
      "authorized-user",
      ...
   ]
}
```

where:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>authUser</td>
<td>Specifies the list of users who have been granted permission to use the On-Premises Connectors in the group.</td>
<td>A comma-separated list of users who have been authorized by the owner of the group. <strong>Required</strong>; can be empty</td>
</tr>
<tr>
<td>data type: Array</td>
<td></td>
<td></td>
</tr>
<tr>
<td>connectorID</td>
<td>The Connector ID of the On-Premises Connector for which information is returned.</td>
<td>The ID for the Connector. Either the Connector ID assigned to the Connector during installation or the label defined for the Connector can be used. <strong>Required</strong></td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Chapter 10: Hybrid Data Pipeline API reference
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectorGroup</td>
<td>Only present if the given connector is a group connector ID.</td>
<td>An object of type <code>connector_group_object</code>, with the following parameters:</td>
</tr>
<tr>
<td>data type: Object</td>
<td></td>
<td>• connectionTimeout: The amount of time, in seconds, that the connectivity service waits for a connection to be established. <code>Required</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• retryDelay: indicates the number of seconds the connectivity service considers the connector disabled. <code>Required</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• loadBalancing: Specifies whether to enable load balancing. <code>Optional</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• members: Specifies the connector ID of each On-Premises Connector in the group, the sequence in which each On-Premises Connector will be tried, and the weight to be applied to the connector. <code>Required</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For more information, see <code>connectorGroup Object</code> on page 1253.</td>
</tr>
<tr>
<td>groups</td>
<td>An object that contains a list of the Connector Groups the Connector is a member of. This object is included in the response only when includeGroups is set to <code>true</code>.</td>
<td>• connectorGroupIds: An array that specifies the Connector Group ID for each group. The array is always present. If the Connector is not a member of any groups, the array is empty.</td>
</tr>
<tr>
<td>owner</td>
<td>The Progress ID of the owner of the Connector. <code>Optional</code></td>
<td>A Progress ID. If <code>owner</code> is specified, its value must match the current owner of the Connector or Connector Group. Changing the owner of a Connector or Connector Group is not supported.</td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sample Server Response

```json
{
    "connectorID": "abcdef01-fedc-abcd-bcde-0123456789ab",
    "owner": "Rick",
    "label": "MyConnector",
    "connectorGroup": {
        "connectionTimeout": 15,
        "retryDelay": 120,
        "loadBalancing": "Round Robin",
        "members": [
            {
                "memberID": "00000000-0000-0000-0000-000000000000",
                "sequence": 1,
                "weight": 1
            }
        ]
    },
    "authUser": [
        "Joe",
        "Fred",
        "Tom"
    ]
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the On-Premises Connector.

**See also**

Get Connectors on page 1230

**Update Connector Information**

**Purpose**

Update the information for an On-Premises Connector. Only the Connector's owner can update the Connector's information. This endpoint can be used to update information for both an individual connector and a connector group.

A group Connector must include a `members` section. Using a `members` section in a simple Connector causes an error.

**URL**

https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>

**Method**

PUT
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<connector-ID> is a unique value associated with the On-Premises Connector. The value is returned using the https://<myserver>:<port>/api/mgmt/connectors GET request.

Request Payload Parameters

The request payload specifies the new definition for the connector. The new definition includes the new Connector Group definition if the connector is a Connector Group and the list of Hybrid Data Pipeline users to add as authorized users. All parameters must be included. The request has the following format:

```
{
    "connectorID": "abcdef01-fedc-abcd-bcde-0123456789ab",
    "owner": "Rick",
    "label": "Development",
    "connectorGroup": {
        "connectionTimeout": 15,
        "retryDelay": 120,
        "loadBalancing": "Round Robin",
        "members": [
            {
                "memberID": "00001000-0001-0001-0010-000010001001",
                "sequence": 1,
                "weight": 1
            },
            {
                "memberID": "00002000-0002-0002-0020-000020002002",
                "sequence": 2,
                "weight": 1
            }
        ]
    },
    "authUser": [
        "Joe",
        "Fred",
        "Tom"
    ]
}
```

where:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>authUser</td>
<td>Specifies the list of users who have been granted permission to use the On-Premises Connectors in the group.</td>
<td>A comma-separated list of users who have been authorized by the owner of the group. <strong>Required</strong>; can be empty</td>
</tr>
<tr>
<td>data type: Array</td>
<td></td>
<td></td>
</tr>
<tr>
<td>connectorID</td>
<td>The Connector ID of the On-Premises Connector for which information is returned.</td>
<td>The ID for the Connector. <strong>Required</strong></td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>connectorGroup</td>
<td>Only present if the given connector is a group connector ID. Group Connectors allow the user to have failover between multiple On-Premises Connectors.</td>
<td></td>
</tr>
<tr>
<td>data type: Object</td>
<td>An object of type connector_group_object, with the following parameters:</td>
<td>- connectionTimeout: The amount of time, in seconds, that the connectivity service waits for a connection to be established. Required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- retryDelay: indicates the number of seconds the connectivity service considers the connector disabled. Required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- loadBalancing: Specifies whether to enable load balancing. Optional.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- members: Specifies the connector ID of each On-Premises Connector in the group, the sequence in which each On-Premises Connector will be tried, and the weight to be applied to the connector. Required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For more information, see connectorGroup Object on page 1240.</td>
</tr>
<tr>
<td>label</td>
<td>A descriptive name for the Connector that can be used instead of the Connector ID. When not specified, the system name is used. Optional</td>
<td>A string with a maximum length of 255 characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If two Connectors in the Group have the same label, the owner's name is appended, for example, Production and Production(SueS).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To delete an existing label, set the value to null.</td>
</tr>
<tr>
<td>owner</td>
<td>The user name of the owner of the Connector. Optional</td>
<td>A user name. If owner is specified, its value must match the current owner of the Connector or Connector Group. Changing the owner of a Connector or Connector Group is not supported.</td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Response Definition

If the Update Connector operation requested is successful, the response is a JSON object defined as:

```json
{
    "connectorID": "connector-id",
    "owner": "owner",
    "label": "label",
    "connectorGroup": {
        "connectionTimeout": 15,
        "retryDelay": 120,
        "loadBalancing": "Round Robin",
        "members": [
            {
                "memberID": "member-id",
                "sequence": 1,
                "weight": 1
            },
            {
                "memberID": "member-id",
                "sequence": 2,
                "weight": 1
            }
        ]
    },
    "authUser": [
        "authorized-user",
        "authorized-user",
        "authorized-user"
    ]
}
```

**Note:** The Update Connector Information response will be the same as the Get Connector Information response. See [Get Connector Information](#) on page 1233.

If the Update Connector operation is not successful, the response is a standard error response.
Sample Success Response

{
   "connectorID": "abcdef01-fedc-abcd-bcde-0123456789ab",
   "owner": "Rick",
   "label": "Production",
   "connectorGroup": {
      "connectionTimeout": 15,
      "retryDelay": 120,
      "loadBalancing": "Round Robin",
      "members": [
         {
            "memberID": "00001000-0001-0001-0010-000010001001",
            "sequence": 1,
            "weight": 1
         },
         {
            "memberID": "00002000-0002-0002-0020-000020002002",
            "sequence": 2,
            "weight": 1
         }
      ]
   }
   "authUser": ["Joe", "Fred", "Tom"]
}

Authentication

Basic Authentication using Login ID and Password.

Authorization

Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the On-Premises Connector.

connectorGroup Object

The connectorGroup object includes parameters that define the way that the Connector group supports connection failover and load balancing.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionTimeout</td>
<td>The amount of time, in seconds, that the connectivity service waits for a connection to be established before timing out the connection request.</td>
<td>Required</td>
<td>0</td>
</tr>
<tr>
<td>data type: String</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>members</td>
<td>Specifies the connector ID of each On-Premise Connector in the group, the sequence in which each On-Premise Connector will be tried, and the weight to be applied to the connector.</td>
<td>Required</td>
<td>An array that modifies the connectorGroup object, with the following parameters: • memberId • sequence • weight For more information, see members Array on page 1255.</td>
</tr>
<tr>
<td>members [memberID,sequence,weight]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>data type: Array [String,int,int]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>retryDelay</td>
<td>When a connection attempt through a particular connector fails, that connector is temporarily disabled in the group to stop connection attempts to a connector that has failed. The value specified for retryDelay indicates the number of seconds the connectivity service considers the connector disabled. The connectivity service does not make any connection attempts. After the retryDelay period has expired, the connector is automatically re-enabled and connection attempts are sent to that connector again. If all of the connectors in a group become disabled at the same time, the connectivity service attempts a connection to each connector in the group instead of suspending all connection attempts until the retryDelay has expired.</td>
<td>Required</td>
<td>$0 \mid x \text{ where } x \text{ is a positive integer that represents a number of seconds.}$ If set to 0, a connection failure for a connector does not disable that connector. If set to $x$, a connection failure to a particular connector will disable that connector for the specified number of seconds. The default is 120.</td>
</tr>
<tr>
<td>loadBalancing</td>
<td>Specifies whether to enable load balancing.</td>
<td>Optional</td>
<td>If set to Round Robin, a round-robin algorithm is used to handle traffic among a group of On-Premises Connectors. If set to null, load balancing is disabled for the Connector group.</td>
</tr>
</tbody>
</table>

**See also**

- [Update Connector Information](#) on page 1236
- [members Array](#) on page 1242

**members Array**

The `members` object is an array that modifies the `connectorGroup` object. The object specifies the connector ID of each On-Premises Connector in the group, the sequence in which each On-Premises Connector will be tried, and the weight to be applied to the connector.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>member_id</td>
<td>Identifies the Connector ID of the On-Premise Connector. It must not be the Connector ID of an On-Premise Connector Group. Nested groups are not supported.</td>
<td>Required</td>
<td>A string comprised of the Connector ID of the On-Premises Connector.</td>
</tr>
<tr>
<td>sequence</td>
<td>For non-load-balanced connector groups, is the relative order in which the On-Premises Connector is tried. The value of the sequence property for each member object must be unique. Duplicate sequence values are not supported and will generate an error response.</td>
<td>Required</td>
<td>0</td>
</tr>
<tr>
<td>weight</td>
<td>For load-balanced connector groups, sets the load for each Connector, with a higher number indicating the relative load directed to the given Connector. For example, if a load-balanced connector group contains connectors A, B and C with weights of 3, 2 and 1 respectively, then for every 6 connections three would go to A, two to B and 1 to C. Moreover, weights do not have to be relative to 1. Rather, they are relative to the other weights in the group. For example, if a group has three connectors with weights 3, 3 and 4, then thirty percent of the requests will go to the first connector, thirty percent will go to the second connector, and forty percent will go to the third connector.</td>
<td>Optional</td>
<td>The default value is 1. Note: For non-load-balanced connector groups, weight is ignored.</td>
</tr>
</tbody>
</table>

See also

connectorGroup Object on page 1240
Get Authorized Users

Purpose
Retrieve the list of users who have been granted permission to use a particular On-Premises Connector.

URL
https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/authuser

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<connector-ID> is a unique value associated with the On-Premises Connector. The value is returned using the <base>/connectors GET request. The authorized user must be the owner of the On-Premises Connector specified.

Response Definition
The response has the following format:

```json
{
  "authUser": [
    <authorized-user>,
    <authorized-user>,
    ...
    <authorized-user>,
  ]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>authUser</td>
<td>Array [String]</td>
<td>Specifies the Hybrid Data Pipeline users who are authorized to use the On-Premises Connector.</td>
<td>authorized-user is a Hybrid Data Pipeline user who is authorized to use the On-Premises Connector.</td>
</tr>
<tr>
<td>[authorized-user]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sample Server Response

```json
{
  "authUser": [
    "Joe",
    "Fred",
    "Tom"
  ]
}
```
**Authentication**
Basic Authentication using Login ID and Password.

**Authorization**
Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the On-Premises Connector.

**Add Authorized Users**

**Purpose**
Add authorized Hybrid Data Pipeline users to an On-Premises Connector’s access control list. The user account can be inactive. If a user name is invalid, an error is returned and none of the specified users are added to the access control list.

**Note:** The list of users is limited to a system-configurable value. If you need to add more authorized users, use additional POST calls to add them. If too many users are provided, an error message that specifies the limit is returned.

**URL**
https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/authuser

**Method**
POST

**URL Parameters**
- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
- `<connector-ID>` is a unique value associated with the On-Premises Connector. The value is returned using the `<base>/connectors` GET request. The authorized user must be the owner of the On-Premises Connector specified.
Request Payload Parameters
The request payload specifies the list of Hybrid Data Pipeline users to add as authorized users. The request has the following format:

```json
{
  "authUser": [
    <authorized-user>,
    <authorized-user>,
    ...
  ]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>authUser</td>
<td>Array [String]</td>
<td>Specifies the Hybrid Data Pipeline users who are authorized to use the On-Premises Connector.</td>
<td>authorized-user is a Hybrid Data Pipeline user who is authorized to use the On-Premises Connector.</td>
</tr>
</tbody>
</table>

Sample Request

```json
{
  "authUser": [
    "Joe",
    "Fred",
    "Tom"
  ]
}
```

Response Definition
If the Add Authorized User operation requested is successful, the response is a JSON object defined as:

```json
{
  "success": true
}
```

If the Add Authorized User operation is not successful, the response is a standard error response.

Sample Server Response

```json
{"success":true}
```

Authentication
Basic Authentication using Login ID and Password.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the On-Premises Connector.
Update Authorized Users

Purpose
Update the list of authorized users for an On-Premises Connector. The list of authorized users specified replaces the current access control list of the specified On-Premises Connector.

URL
https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/authuser

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<connector-ID> is a unique value associated with the On-Premises Connector. The value is returned using the <base>/connectors GET request. The authorized user must be the owner of the On-Premises Connector specified.

Request Payload Parameters
The request payload specifies the list of Hybrid Data Pipeline users to add as authorized users. The request has the following format:

```json
{   "authUser": [     <authorized-user>,     <authorized-user>,     ...     <authorized-user>   ]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>authUser</td>
<td>Array [String]</td>
<td>Specifies the Hybrid Data Pipeline users who are authorized to use the On-Premises Connector.</td>
<td>authorized-user is a Hybrid Data Pipeline user who is authorized to use the On-Premises Connector.</td>
</tr>
<tr>
<td>[authorized-user]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Response Definition
If the Update Authorized User operation requested is successful, the response is a JSON object defined as:

```json
{   "success":true
}
```

If the Update Authorized User operation is not successful, the response is a standard error response.
Sample Success Response

{"success":true}

Authentication
Basic Authentication using Login ID and Password.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the On-Premises Connector.

Delete Authorized Users
Providing a body with the DELETE method is not forbidden by the HTTP specifications. However, many HTTP libraries either do not allow or do not work correctly when a body is specified in a DELETE request.

WORKAROUND: To delete one or more authorized users from an On-Premise Connector, issue a PUT request to the /connectors/<connector-ID>/authuser endpoint and remove the users to be deleted from the authuser array in the request payload.

Purpose
Revoke permission to use this On-Premises Connector from some or all users. If a user name is invalid, an error is returned and no user permissions are revoked.

Note: The list of users for which permission to use the connector is revoked is limited to a system-configurable value. If you need to delete more than the system limit, use additional DELETE calls to add them. If too many users are provided, an error message that specifies the limit is returned.

URL
https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/authuser

Method
DELETE

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<connector-ID> is a unique value associated with the On-Premises Connector. The value is returned using the <base>/connectors GET request. The authorized user must be the owner of the On-Premises Connector specified.
Request Payload Parameters

The request payload specifies the list of Hybrid Data Pipeline users to remove from the On-Premise Connector access control list. The request has the following format:

```json
{
  "authUser": [
    <authorized-user>,
    <authorized-user>,
    ...
    <authorized-user>
  ]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>authUser</td>
<td>Array [String]</td>
<td>Specifies the Hybrid Data Pipeline users who are authorized to use the On-Premises Connector.</td>
<td>authorized-user is a Hybrid Data Pipeline user who is authorized to use the On-Premises Connector.</td>
</tr>
<tr>
<td>[authorized-user]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sample Server Request

```json
{
  "authUser": [
    "Joe",
    "Fred",
    "Tom"
  ]
}
```

Response Definition

If the Remove Authorized User operation requested is successful, the response is a JSON object defined as

```json
{
  "success": true
}
```

If the Remove Authorized User operation is not successful, the response is a standard error response.

Authentication

Basic Authentication using Login ID and Password.

Authorization

Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the On-Premises Connector.

Create a Connector Group

Purpose

Creates a group of On-Premises Connectors. The group can be used to support failover and load balancing across two or more On-Premises Connectors.
**Note:** An On-Premises Connector can be a member of only one Group. If you specify a ConnectorID that is a member of another group, the connectivity service returns an error, and the Connector is not added to the GroupConnector.

**URL**

https://<myserver>:<port>/api/mgmt/connectors

**Method**

POST

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Request Payload Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectorGroup</td>
<td>An object that contains a group of On-Premise connectors that can be used to support failover and load balancing across two or more On-Premises Connectors.</td>
<td>An object of type connector_group_object, with the following parameters:</td>
</tr>
<tr>
<td>data type: Object</td>
<td></td>
<td>• connectionTimeout: The amount of time, in seconds, that the connectivity service waits for a connection to be established. <strong>Required.</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• retryDelay: indicates the number of seconds the connectivity service considers the connector disabled. <strong>Required.</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• loadBalancing: Specifies whether to enable load balancing. <strong>Optional.</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• members: Specifies the connector ID of each On-Premises Connector in the group, the sequence in which each On-Premises Connector will be tried, and the weight to be applied to the connector. <strong>Required.</strong></td>
</tr>
</tbody>
</table>

For more information, see connectorGroup Object on page 1253.
Sample Request Payload

This request creates a new group connector to be owned by the current user comprised of three member On-Premises Connectors. Each member connector must already be registered to the owner.

```json
{
  "owner": "Rick",
  "label": "Production",
  "connectorGroup": {
    "connectionTimeout": 15,
    "retryDelay": 120,
    "loadBalancing": "Round Robin",
    "members": [
      {
        "memberID": "00000000-0000-0000-0011-000000000010",
        "sequence": 1,
        "weight": 60
      },
      {
        "memberID": "00021111-0011-0011-0022-000000222200",
        "sequence": 2,
        "weight": 20
      },
      {
        "memberID": "00031313-0011-0011-0033-000000333300",
        "sequence": 3,
        "weight": 20
      }
    ],
    "authUser": [
      "Joe",
      "Fred",
      "Tom"
    ]
  }
}
```

Response Definition

The response has the following format:

```json
{
  "connectorID": "<group-connector-id>",
  "owner": "<owner-user-id>",
  "label": "<label>",
  "connectorGroup": {
    "connectionTimeout": 15,
    "retryDelay": 120,
    "loadBalancing": "Round Robin",
    "members": [
      {
        "memberID": "<member-id>",
        "sequence": 1,
        "weight": 60
      },
      {
        "memberID": "<member-id>",
        "sequence": 2,
        "weight": 20
      },
      {
        "memberID": "<member-id>",
        "sequence": 3,
        "weight": 20
      }
    ],
    "authUser": [
      "Joe",
      "Fred",
      "Tom"
    ]
  }
}
```
Sample Server Response

After sending in the payload successfully, the server response includes the above information and a newly generated ConnectorID for the On-Premise Connector Group. The user is also assigned as the owner of the Connector Group.

**Note:** The newly generated ConnectorID is also referred to as a Connector Group ID. You must specify this Connector Group ID in a data source to implement failover and load balancing.

```
{
  "connectorID": "12345678-90ab-cdef-ghij-klmnopqrstuvwxyz",
  "owner": "Rick",
  "label": "Production",
  "connectorGroup": {
    "connectionTimeout": 15,
    "retryDelay": 120,
    "loadBalancing": "Round Robin",
    "members": [
      {
        "memberID": "00000000-0000-0000-0011-000000000010",
        "sequence": 1,
        "weight": 60
      },
      {
        "memberID": "00021111-0011-0011-0022-000002222200",
        "sequence": 2,
        "weight": 20
      },
      {
        "memberID": "00031313-0011-0011-0033-000003333300",
        "sequence": 3,
        "weight": 20
      }
    ]
  },
  "authUser": [
    "Joe",
    "Fred",
    "Tom"
  ]
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

Any active Hybrid Data Pipeline user

**See also**

- connectorGroup Object on page 1253
- members Array on page 1255
connectorGroup Object

The connectorGroup object includes parameters that define the characteristics of the connector group, including the way that the connector group supports connection failover and load balancing.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
</table>
| connectionTimeout     | The amount of time, in seconds, that the connectivity service waits for a connection to be established before timing out the connection request. | Required | 0 | x where x is a positive integer that represents a number of seconds.  
If set to 0, the connectivity service does not time out a connection request.  
If set to x, the connectivity service waits for the specified number of seconds before giving up on the current connection attempt and attempting to establish a connection through the next connector in the group. If all of the connectors in the group are tried without establishing a connection, control is returned to the application and a timeout error is generated.  
The default is 15. |
| data type: String     |                                                                                                                            |        |                                                                                                                                              |
| members               | Specifies the connector ID of each On-Premise Connector in the group, the sequence in which each On-Premise Connector will be tried, and the weight to be applied to the connector. | Required | An array that modifies the connectorGroup object, with the following parameters:  
• memberID  
• sequence  
• weight  
For more information, see members Array on page 1255. |
| data type: Array [String,int,int] |                                                                                                                            |        |                                                                                                                                              |
### retryDelay

**Description**
When a connection attempt through a particular connector fails, that connector is temporarily disabled in the group to stop connection attempts to a connector that has failed. The value specified for retryDelay indicates the number of seconds the connectivity service considers the connector disabled. The connectivity service does not make any connection attempts. After the retryDelay period has expired, the connector is automatically re-enabled and connection attempts are sent to that connector again. If all of the connectors in a group become disabled at the same time, the connectivity service attempts a connection to each connector in the group instead of suspending all connection attempts until the retryDelay has expired.

**Usage**
Required

**Valid Values**
0 | \(x\) where \(x\) is a positive integer that represents a number of seconds. If set to 0, a connection failure for a connector does not disable that connector. If set to \(x\), a connection failure to a particular connector will disable that connector for the specified number of seconds. The default is 120.

### loadBalancing

**Description**
Specifies whether to enable load balancing.

**Usage**
Optional

**Valid Values**
If set to Round Robin, a round-robin algorithm is used to handle traffic among a group of On-Premises Connectors. Omit the loadBalancing property to disable load balancing for the Connector Group.

### See also
- Create a Connector Group on page 1249
- members Array on page 1255

### members Array
The members object is an array that modifies the connectorGroup object. The object specifies the connector ID of each On-Premises Connector in the group, the sequence in which each On-Premises Connector will be tried, and the weight to be applied to the connector.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>member_id</td>
<td>Identifies the Connector ID of the On-Premise Connector. It must not be the Connector ID of an On-Premise Connector Group. Nested groups are not supported.</td>
<td>Required</td>
<td>A string comprised of the Connector ID of the On-Premises Connector.</td>
</tr>
<tr>
<td>sequence</td>
<td>For non-load-balanced connector groups, is the relative order in which the On-Premises Connector is tried. The value of the sequence property for each member object must be unique. Duplicate sequence values are not supported and will generate an error response.</td>
<td>Required</td>
<td>0</td>
</tr>
<tr>
<td>weight</td>
<td>For load-balanced connector groups, sets the load for each Connector, with a higher number indicating the relative load directed to the given Connector. For example, if a load-balanced connector group contains connectors A, B and C with weights of 3, 2 and 1 respectively, then for every 6 connections three would go to A, two to B and 1 to C. Moreover, weights do not have to be relative to 1. Rather, they are relative to the other weights in the group. For example, if a group has three connectors with weights 3, 3 and 4, then thirty percent of the requests will go to the first connector, thirty percent will go to the second connector, and forty percent will go to the third connector.</td>
<td>Optional</td>
<td>The default value is 1.</td>
</tr>
</tbody>
</table>

**Note:** For non-load-balanced connector groups, weight is ignored.

**See also**

[connectorGroup Object on page 1253](#)
Add On-Premises Connectors to an On-Premises Connector Group

Purpose
Adds specified On-Premises Connectors to a group of On-Premise Connectors, and specifies the order in which each On-Premises Connector is tried in a failover scenario.

Note: An On-Premises Connector can be a member of only one Group. If you specify a ConnectorID that is already in use, the connectivity service returns an error, and the Connector is not added to the GroupConnector.

URL
https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/members

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
### Payload Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>members [memberID, sequence, weight]</td>
<td>Array [String,int,int]</td>
<td>Specifies the connector ID of each On-Premise Connector in the group, the sequence in which each On-Premise Connector will be tried, and the weight to be applied to the connector.</td>
<td>Required</td>
<td>memberID is the Connector ID of the On-Premises Connector. It must not be the Connector ID of an On-Premises Connector Group. Nested groups are not supported. sequence, for non-load-balanced connector groups, is the relative order in which the On-Premise Connector is tried. The value of the sequence property for each member object must be unique. Duplicate sequence values are not supported and will generate an error response. weight, for load-balanced connector groups, sets the load for each Connector, with a higher number indicating the relative load directed to the given Connector. For example, if a load-balanced connector group contains connectors A, B and C with weights of 3, 2 and 1 respectively, then for every 6 connections three would go to A, two to B and 1 to C. Moreover, weights do not have to be relative to 1. Rather, they are relative to the other weights in the group. For example, if a group has three connectors with weights 3, 3 and 4, then thirty percent of the requests will go to the first connector, thirty percent will go to the second connector, and forty percent will go to the third connector. The default value is 1.</td>
</tr>
</tbody>
</table>
Sample Request Payload

This request adds three On-Premises Connectors to an existing GroupConnector that contained only one On-Premises Connector.

```
{
   "members": [
   {
      "memberID": "00000000-0000-0000-0044-000000000040",
      "sequence": 2,
      "weight": 4
   },
   {
      "memberID": "00021111-0011-0011-0055-000000555500",
      "sequence": 3,
      "weight": 3
   },
   {
      "memberID": "00061616-0011-0011-0063-000000363600",
      "sequence": 4,
      "weight": 2
   }
   ]
}
```

Response Definition

The response has the following format:

```
{ "members": [
   {
      "memberID": "<memberID>",
      "sequence": "<sequence>",
      "weight": "<weight>"
   },
   {
      "memberID": "<memberID>",
      "sequence": "<sequence>",
      "weight": "<weight>"
   },
   {
      "memberID": "<memberID>",
      "sequence": "<sequence>",
      "weight": "<weight>"
   }
] }
Sample Server Response

After sending in the payload, on success, the owner of the group connector receives a response with the following information.

```json
{
    "members": [
        {
            "memberID": "00000001-0000-0000-0001-000000000111",
            "sequence": 1,
            "weight": 1
        },
        {
            "memberID": "00000000-0000-0000-0044-000000000040",
            "sequence": 2,
            "weight": 4
        },
        {
            "memberID": "00021111-0011-0011-0055-000000555500",
            "sequence": 3,
            "weight": 3
        },
        {
            "memberID": "00061616-0011-0011-0063-000000363600",
            "sequence": 4,
            "weight": 2
        }
    ]
}
```

Authentication

Basic Authentication using Login ID and Password.

Authorization

Any active Hybrid Data Pipeline user

Get the List of On-Premises Connectors in an On-Premises Connector Group

Purpose

Retrieve the list of On-Premises Connectors in an On-Premises Connector group.

URL

https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/members

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
<connector-ID> is a unique value associated with the group On-Premises Connector. The value is returned using the <base>/connectors/GET request. The authorized user must be the owner of the group On-Premises Connector specified.
Response Definition
The response has the following format:

```
{
    "members": [
        {
            <memberID>,
            <sequence>,
            <weight>
        },
        {
            <memberID>,
            <sequence>,
            <weight>
        },
        {
            <memberID>,
            <sequence>,
            <weight>
        }
    ]
}
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>members</td>
<td>Array [String]</td>
<td>Specifies the connector ID of each On-Premises Connector in the group.</td>
<td><strong>memberID</strong> is the Connector ID of the On-Premises Connector. It must not be the Connector ID of an On-Premises Connector Group. Nested groups are not supported. <strong>sequence</strong>, for non-load-balanced connector groups, is the relative order in which the On-Premise Connector is tried. The value of the sequence property for each member object must be unique. Duplicate sequence values are not supported and will generate an error response. <strong>weight</strong>, for load-balanced connector groups, sets the load for each Connector, with a higher number indicating the relative load directed to the given Connector. For example, if a load-balanced connector group contains connectors A, B and C with weights of 3, 2 and 1 respectively, then for every 6 connections three would go to A, two to B and 1 to C. Moreover, weights do not have to be relative to 1. Rather, they are relative to the other weights in the group. For example, if a group has three connectors with weights 3, 3 and 4, then thirty percent of the requests will go to the first connector, thirty percent will go to the second connector, and forty percent will go to the third connector. The default value is 1. <strong>Note</strong>: <strong>weight</strong> is optional. For non-load-balanced connector groups, <strong>weight</strong> is ignored.</td>
</tr>
</tbody>
</table>

Note: **weight** is optional. For non-load-balanced connector groups, **weight** is ignored.
Sample Server Response

```json
{
  "members": [
    {
      "memberID": "00001110-0000-0000-0000-000000001111",
      "sequence": 1,
      "weight": 1
    },
    {
      "memberID": "00002220-0000-0000-0000-000000002222",
      "sequence": 2,
      "weight": 2
    }
  ]
}
```

Authentication
Basic Authentication using Login ID and Password.

Authorization
Any active Hybrid Data Pipeline user. The authenticated user must be the owner of the group On-Premises Connector.

Configure Round-Robin Request Balancing for an On-Premises Connector Group

Purpose
Specify the On-Premises Connectors in a connector group, and define a weight for each.

Note: A connector can be used in only one group, and only once in that group. Each member connector must already be registered to the owner.

URL
https://<myserver>:<port>/api/mgmt/connectors/<group-connector>

Method
PUT

URL Parameters
- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
- `<group-connector>` is a unique value associated with the On-Premises Connector. The value is returned using the `<base>/connectors/` GET request.
## Request Payload Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionTimeout</td>
<td>String</td>
<td>The amount of time, in seconds, that the connectivity service waits for a connection to be established before timing out the connection request.</td>
<td>Optional</td>
<td>$0 \mid x$ where $x$ is a positive integer that represents a number of seconds. If set to 0, the connectivity service does not time out a connection request. If set to $x$, the connectivity service waits for the specified number of seconds before giving up on the current connection attempt and attempting to establish a connection through the next connector in the group. If all of the connectors in the group are tried without establishing a connection, control is returned to the application and a timeout error is generated. The default is 15.</td>
</tr>
</tbody>
</table>
### retryDelay

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>retryDelay</td>
<td>String</td>
<td>When a connection attempt through a particular connector fails, that connector is temporarily disabled in the group to stop connection attempts to a connector that has failed. The value specified for retryDelay indicates the number of seconds the connectivity service considers the connector disabled. The connectivity service does not make any connection attempts. After the retryDelay period has expired, the connector is automatically re-enabled and connection attempts are sent to that connector again. If all of the connectors in a group become disabled at the same time, the connectivity service attempts a connection to each connector in the group instead of suspending all connection attempts until the retryDelay has expired.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Optional</td>
<td>0</td>
<td>x where x is a positive integer that represents a number of seconds. If set to 0, the connectivity service does not delay between retries. If set to x, the connectivity service waits between connection retry attempts the specified number of seconds. The default is 120.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Data Type</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>loadBalancing</td>
<td>String</td>
<td>Specifies whether to balance requests across multiple On-Premises Connectors in a connector group.</td>
<td>Optional</td>
<td>If set to <strong>Round Robin</strong>, a round-robin algorithm is used to handle traffic among a group of On-Premises Connectors.</td>
</tr>
<tr>
<td>members</td>
<td>Array [String,int,int]</td>
<td>Specifies the connector ID of each On-Premise Connector in the group, the sequence in which each On-Premise Connector will be tried, and the weight to be applied to the connector.</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Data Type</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>-------------</td>
<td>-------</td>
<td>--------------</td>
</tr>
</tbody>
</table>

memberID is the Connector ID of the On-Premises Connector. It must not be the Connector ID of an On-Premises Connector Group. Nested groups are not supported.

sequence, for non-load-balanced connector groups, is the relative order in which the On-Premise Connector is tried. The value of the sequence property for each member object must be unique. Duplicate sequence values are not supported and will generate an error response.

weight, for load-balanced connector groups, sets the load for each Connector, with a higher number indicating the relative load directed to the given Connector. For example, if a load-balanced connector group contains connectors A, B and C with weights of 3, 2 and 1 respectively, then for every 6 connections three would go to A, two to B and 1 to C. Moreover, weights do not have to be relative to 1. Rather, they are relative to the other weights in the group. For example, if a group has three connectors with weights 3, 3 and 4, then thirty percent of the requests will go to the first connector, thirty percent will go to the second connector, and forty percent will go to the third connector. The default value is 1.

Note: weight is optional. For non-load-balanced connector groups, weight is ignored.
Sample Request Payload

This request defines the order in which the On-Premises Connectors are tried when load balancing is enabled. In this example, we have three Connectors with weights (relative performance settings) of 3, 2 and 1. For every six connections, three will go to the Connector with weight=3, two will go to the Connector with weight=2, and one to the Connector with weight=1. If a connection attempt fails, different Connectors will be tried until either a connection succeeds or all Connectors have failed to connect.

```json
{
    "owner": "Rick",
    "label": "DevGroup1",
    "connectorGroup": {
        "connectionTimeout": "15",
        "retryDelay": "120",
        "loadBalancing": "Round Robin",
        "members": [
            {
                "memberID": "00000000-0000-0000-0011-000000000010",
                "sequence": 3,
                "weight": 1
            },
            {
                "memberID": "00021111-0011-0011-0022-0000222200",
                "sequence": 2,
                "weight": 2
            },
            {
                "memberID": "00031313-0011-0011-0033-0000333300",
                "sequence": 1,
                "weight": 3
            }
        ]
    },
    "authUser": [
        "Joe",
        "Fred",
        "Tom"
    ]
}
```

Response Definition

The response has the following format:

```json
{
    "connectorID": "<group-connector-id>",
    "owner": "owner",
    "label": "<label>",
    "connectorGroup": {
        "connectionTimeout": "15",
        "retryDelay": "120",
        "loadBalancing": "Round Robin",
        "members": [
            {
                "memberID": "<memberID>",
                "sequence": 3,
                "weight": 1
            },
            {
                "memberID": "<memberID>",
                "sequence": 2,
                "weight": 2
            },
            {
                "memberID": "<memberID>",
```
Sample Server Response

After sending in the payload, on success, the user Rick receives a response with the preceding information plus the newly generated ConnectorID, and is assigned as the owner.

```
{
  "connectorID": "12345678-90ab-cdef-ghij-kl123pqrstuv",
  "owner": "Rick",
  "label": "DevGroup1",
  "connectorGroup": {
    "connectionTimeout": 15,
    "retryDelay": 120,
    "loadBalancing": "Round Robin",
    "members": [
      {
        "memberID": "00000000-0000-0000-0011-000000000010",
        "sequence": 3,
        "weight": 1
      },
      {
        "memberID": "00021111-0011-0011-0022-000000222200",
        "sequence": 2,
        "weight": 2
      },
      {
        "memberID": "00031313-0011-0011-0033-000000333300",
        "sequence": 1,
        "weight": 3
      }
    ],
  },
  "authUser": [
    "Joe",
    "Fred",
    "Tom"
  ]
}
```

Authentication

Basic Authentication using Login ID and Password.

Authorization

Any active Hybrid Data Pipeline user

Related Topics

- Create a Connector Group on page 1249
- Configuring Failover and Balancing Requests with an On-Premises Connector Group on page 1230
Replace the List of On-Premises Connectors in an On-Premises Connector Group

Purpose
Replaces the current set of members of the group with the set of members specified in the request payload.

URL
https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/members

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<connector-ID> is a unique value associated with the group On-Premises Connector. The value is returned using the <base>/connectors/GET request. The authorized user must be the owner of the group On-Premises Connector specified.
### Payload Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>members</td>
<td>Array [String, int, int]</td>
<td>Specifies the connector ID of each On-Premise Connector in the group, the sequence in which each On-Premise Connector will be tried, and the weight to be applied to the connector.</td>
<td>Required</td>
<td>memberID is the Connector ID of the On-Premises Connector. It must not be the Connector ID of an On-Premises Connector Group. Nested groups are not supported. Sequence, for non-load-balanced connector groups, is the relative order in which the On-Premise Connector is tried. The value of the sequence property for each member object must be unique. Duplicate sequence values are not supported and will generate an error response. Weight, for load-balanced connector groups, sets the load for each Connector, with a higher number indicating the relative load directed to the given Connector. For example, if a load-balanced connector group contains connectors A, B and C with weights of 3, 2 and 1 respectively, then for every 6 connections three would go to A, two to B and 1 to C. Moreover, weights do not have to be relative to 1. Rather, they are relative to the other weights in the group. For example, if a group has three connectors with weights 3, 3 and 4, then thirty percent of the requests will go to the first connector, thirty percent will go to the second connector, and forty percent will go to the third connector. The default value is 1.</td>
</tr>
</tbody>
</table>

**Note:** weight is optional. For non-load-balanced connector groups, weight is ignored.

### Sample Request Payload

The groups list of On-Premises Connectors is replaced with the following list of three connectors.

```json
[]
```
"members": [  
  {  
    "memberID": "00000000-0000-0000-0044-000000000040",  
    "sequence": 1,  
    "weight": 2  
  },  
  {  
    "memberID": "00021111-0011-0011-0055-000000555500",  
    "sequence": 2,  
    "weight": 3  
  },  
  {  
    "memberID": "00061616-0011-0011-0063-000000363600",  
    "sequence": 3,  
    "weight": 1  
  }  
]}

This request replaces the groups list of On-Premises Connectors with the following list of two connectors. The net effect of this request is that it removes the second On-Premises Connector and adjusts the sequence of the next member On-Premises Connector.

{
  "members": [  
    {  
      "memberID": "00000000-0000-0000-0044-000000000040",  
      "sequence": 1,  
      "weight": 2  
    },  
    {  
      "memberID": "00061616-0011-0011-0063-000000363600",  
      "sequence": 2,  
      "weight": 1  
    }  
  ]
}

Response Definition

The response has the following format:

{
  "members": [  
    {  
      "memberID": "<memberID>",  
      "sequence": "<sequence>",  
      "weight": "<weight>"  
    },  
    {  
      "memberID": "<memberID>",  
      "sequence": "<sequence>",  
      "weight": "<weight>"  
    },  
    {  
      "memberID": "<memberID>",  
      "sequence": "<sequence>",  
      "weight": "<weight>"  
    }  
  ]
}
Sample Response Payload
After sending in the payload, on success, the owner of the group connector receives a response with the following information.

```json
{
  "members": [
    {
      "memberID": "00000000-0000-0000-0044-000000000040",
      "sequence": 1,
      "weight": 1
    },
    {
      "memberID": "00061616-0011-0011-0063-000000363600",
      "sequence": 2,
      "weight": 2
    }
  ]
}
```

Authentication
Basic Authentication using Login ID and Password.

Authorization
Any active Hybrid Data Pipeline user

Remove an On-Premises Connector
DEPRECATED. Providing a body with the DELETE method is not forbidden by the HTTP specifications. However, many HTTP libraries either do not allow DELETE requests or do not work correctly when a body is specified in a DELETE request.

WORKAROUND: Either the Replace the List of On-Premises Connectors endpoint or the Update Connector Information endpoint can be used to remove members from a group.

Purpose
Remove an On-Premises Connector from an On-Premises Connector Group. Optionally, you can also delete the group's Connector ID.

To delete one or more connectors from a connector group, issue a PUT request to the /connectors/<connector-ID>/members endpoint, and remove the connectors to be deleted from the members array in the request payload.

The authorized user must be the owner of the On-Premises Connector specified.

Note: You cannot remove all On-Premises Connectors in an On-Premise Connector group. To delete the group, use the Delete Group API.

URL
https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>/members

Method
DELETE
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<connector-ID> is a unique value associated with the group On-Premises Connector. The value is returned using the <base>/connectors/GET request. The authorized user must be the owner of the group On-Premises Connector specified.

Request Payload Parameters

The request payload specifies the list of Hybrid Data Pipeline users to remove from the On-Premise Connector group. The request has the following format:

```json
{
   "members": [<memberID>, <memberID>]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>members</td>
<td>Array [String]</td>
<td>Specifies the connector ID of each On-Premises Connector in the group.</td>
<td>memberID is the Connector ID of the On-Premises Connector. It must not be the Connector ID of an On-Premises Connector Group. Nested groups are not supported.</td>
</tr>
</tbody>
</table>

Sample Server Request

```json
{
   "member": ["00021111-0011-0011-0022-000000222200", "00031313-0011-0011-0033-000000333300"]
}
```

Response Definition

If the Remove On-Premises Connectors operation requested is successful, the response is a JSON object defined as

```json
{"success":true}
```

If the Remove On-Premises Connectors operation is not successful, the response is a standard error response.

Authentication

Basic Authentication using Login ID and Password.
Authorization

Only the owner of the On-Premises Connector can remove member On-Premises Connectors from the On-Premises Connector Group.

Delete a Group

Purpose

Delete an On-Premises Connector Group.

The authorized user must be the owner of the On-Premises Connector specified.

URL

https://<myserver>:<port>/api/mgmt/connectors/<connector-ID>

Method

DELETE

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<connector-ID> is a unique value associated with the group On-Premises Connector. The value is returned using the <base>/connectors/GET request. The authorized user must be the owner of the group On-Premises Connector specified.

Response Definition

If the Remove On-Premises Connectors operation requested is successful, the response is a JSON object defined as

```
{
   "success":true
}
```

If the Remove Group operation is not successful, the response is a standard error response.

Authentication

Basic Authentication using Login ID and Password.

Authorization

Only the owner of the On-Premises Connector can delete the On-Premise Connector Group.
Data Sources API

Hybrid Data Pipeline enables access to a variety of data stores, such as Apache Hive, DB2, SQL Server, Oracle, and Salesforce. To access data residing on a backend data store, administrators or users must create a Hybrid Data Pipeline data source. A Hybrid Data Pipeline data source can be created by specifying parameters associated with a specific data store. The information provided in the data source allows the service to connect to the backend data store. A data source can be created with the Web UI or the Data Sources API.

Foremost, the Data Sources API enables users to create data sources. A user must have the CreateDataSource (1) permission to create a data source. When a user creates a data source, he or she is the owner of the data source. In turn, data source owners can view, modify, and delete data sources they own, if they have the corresponding permissions for these operations. For example, a data source owner must have the ViewDataSource (2) permission to view the data source, and the ModifyDataSource (3) permission to modify the data source.

Note: The Schema API on page 1412 and the Driver Files API on page 1360 are extensions of the Data Sources API. The Schema API can be used to retrieve the information needed to configure a schema for OData connectivity. The Drive Files API can be used to retrieve and manage files used to support data connectivity to non-relational data stores and REST services.

The Data Sources API also supports advanced functionality that allows data source owners to share data sources with other users and enables administrators to create and manage data sources on behalf of users. See the following topics for more information.

- Sharing data sources on page 1279
- Managing resources on behalf of users on page 1281

The following table lists the operations that can be performed using the Data Sources API.

<table>
<thead>
<tr>
<th>Task</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve a list of available data stores and their options</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datastores</td>
</tr>
<tr>
<td>Retrieve the details for a particular data store</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datastores/{datastoreId}</td>
</tr>
<tr>
<td>Create a data source or group data source</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources</td>
</tr>
<tr>
<td>Retrieve a list of data sources</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources</td>
</tr>
<tr>
<td>Retrieve the details for a data source</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}</td>
</tr>
<tr>
<td>Update the options and values for a data source</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}</td>
</tr>
<tr>
<td>Delete a data source</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}</td>
</tr>
<tr>
<td>Task</td>
<td>Request</td>
<td>URL</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Retrieve permissions on a data source</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/permissions</td>
</tr>
<tr>
<td>Update permissions on data sources</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/permissions</td>
</tr>
<tr>
<td>Test a connection to a data source</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/test</td>
</tr>
<tr>
<td>Refresh the cached object mapping of a data source</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/map</td>
</tr>
<tr>
<td>Create or refresh a data source OData model</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/model</td>
</tr>
<tr>
<td>Check status of the OData model refresh</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/model</td>
</tr>
<tr>
<td>Retrieve the members of a group data source</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{groupDatasourceId}/members</td>
</tr>
<tr>
<td>Add member data sources to a group data source</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{groupDatasourceId}/members</td>
</tr>
<tr>
<td>Update members of a group data source</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{groupDatasourceId}/members</td>
</tr>
<tr>
<td>Delete a member data source from a group data source</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{groupDatasourceId}/members/{memberDatasourceId}</td>
</tr>
<tr>
<td>Retrieve users with whom data source is being shared</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/sharedUsers</td>
</tr>
<tr>
<td>Share data source with a user or users</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/sharedUsers</td>
</tr>
<tr>
<td>Stop sharing the data source with users</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/sharedUsers</td>
</tr>
<tr>
<td>Retrieve the data source permissions for a user with whom the data source is being shared</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/sharedUsers/{userId}</td>
</tr>
<tr>
<td>Update the data source permissions for a user with whom the data source is being shared</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceId}/sharedUsers/{userId}</td>
</tr>
</tbody>
</table>
### Sharing data sources

The Data Sources API allows users to share data sources. When a user creates a data source, he or she is the owner of the data source. A data source can be shared with either Hybrid Data Pipeline user accounts or tenants. Either administrators or standard users can share data sources with other users, but only administrators can share data sources with tenants. As the following sections show, most rules that govern data source sharing depend on whether the data source is being shared with user accounts or tenants.

- **General notes and guidelines**
- **Sharing data sources with Hybrid Data Pipeline user accounts**
- **Sharing data sources with Hybrid Data Pipeline tenants**

### General notes and guidelines

- When a data source is shared with a tenant, the data source is in effect shared with all users in the tenant. However, a data source cannot be shared simultaneously with a tenant and users in the same tenant. When a data source is first shared with users in a tenant and subsequently shared with the same tenant, the shared users are removed from the data source. These individual users will still be able to use the shared data source but only through the share made to the tenant. In turn, once a data source has been shared with a tenant, the data source cannot subsequently be shared with users in the same tenant.
- A user with whom a data source has been shared can be moved from one tenant to another. If the owner of the data source is an administrator of the target tenant, the user will continue to have access to the shared data source. However, if the owner is not an administrator of the target tenant, the user will no longer have access to the data source.
• Sharing a data source group requires that the member data sources of the group also be shared.

• Data source groups may only be created with member data sources that are owned by the creator. In other words, the creator of a data source group cannot include a data source shared by another user in the data source group he or she is creating.

Sharing data sources with Hybrid Data Pipeline user accounts

• Either administrators or standard users can share data sources with other users.

• To share a data source with a tenant, the data source owner must have either set of the following permissions.
  
  • The Administrator (12) permission.
  
  • The MgmtAPI (11) permission, the ModifyDataSource (3) permission, and administrative access on the tenant with which the data source is being shared.

• The data source owner must apply permissions to the data source. The following permissions can be applied to data sources: ViewDataSource (2), ModifyDataSource (3), UseDataSourceWithJDBC (5), UseDataSourceWithODC (6), and UseDataSourceWithOData (7). For example, a data source owner may want to share a data source with another user but limit the user's access to OData queries. Therefore, the data source owner would grant only the UseDataSourceWithOData (7) permission to the user.

• A data source owner cannot apply permissions to shared data sources he or she does not have. Similarly, an administrator sharing a data source on behalf of the owner cannot apply permissions which the owner does not have.

• The data source owner can share the data source with any administrator of the tenant to which he or she belongs and with other users in the tenant to which he or she belongs.

• A tenant administrator – a user with administrative access to one or more tenants – can share a data source he or she has created with users in tenants he or she administers.

• A system administrator – a user with the Administrator (12) permission – can share a data source he or she has created with any user in any tenant.

• A shared data source cannot be deleted. The data source owner must stop sharing the data source with users before the data source can be deleted.

• A shared data source owner cannot be deleted. The user accounts with which the data source is being shared must be removed before the shared data source owner can be deleted.

• A shared data source owner cannot be moved from one tenant to another. The data source owner must stop sharing the data source before he or she can be moved.

• A shared data source cannot be renamed.

• A data source cannot be shared with a user account that already has a data source with the same name.

Sharing data sources with Hybrid Data Pipeline tenants

• Only administrators can share data sources with tenants.

• A tenant administrator – a user with administrative access to one or more tenants – can share a data source he or she has created with any tenant he or she administers.

• A system administrator – a user with the Administrator (12) permission – can share a data source he or she has created with any tenant.

• The administrator owner of the data source must have either the Administrator (12) permission; or the MgmtAPI (11) permission, the ModifyDataSource (3) permission, and administrative access on any tenant with which the data source will be shared.
• The administrator owner of the data source must apply permissions to the data source. The following permissions can be applied to shared data sources: ViewDataSource (2), ModifyDataSource (3), UseDataSourceWithJDBC (5), UseDataSourceWithODBC (6), and UseDataSourceWithOData (7). For example, a data source owner may want to share a data source with another user but limit the user's access to OData queries. Therefore, the data source owner would grant only the UseDataSourceWithOData (7) permission to the user.

• The administrator owner of the data source cannot apply permissions to shared data sources he or she does not have. Similarly, an administrator sharing a data source on behalf of the owner cannot apply permissions which the owner does not have.

• A shared data source cannot be deleted. The administrator owner of the data source must stop sharing the data source with users and tenants before the data source can be deleted.

• The administrator owner of a shared data source cannot be deleted. The user accounts and tenants with which the data source is being shared must be removed before the administrator owner can be deleted.

• The administrator owner of a shared data source cannot be moved from one tenant to another. The data source owner must stop sharing the data source before he or she can be moved.

• A shared data source cannot be renamed.

• A data source cannot be shared with a tenant if any user account in the tenant already has a data source with the same name.

See also
User provisioning on page 107

Managing resources on behalf of users

The Hybrid Data Pipeline API allows administrators to manage several resources on behalf of users. Administrators can carry out a number of API operations by passing the name of a user account with the ?user query parameter. For example, the following query retrieves a list of data sources on behalf of the TestUser user account.

GET https://<myserver>:<port>/api/mgmt/datasources?user=TestUser

System administrators need no permissions beyond the Administrator (12) permission to execute operations on behalf of any user across the system, including users that reside in different tenants. However, administrators who do not have the Administrator (12) permission must meet the following criteria to execute operations on behalf of users.

• Tenant-level administrators (administrators who reside in a tenant other than the default system tenant) must belong to the same tenant to which the user belongs. System-level administrators (administrators who reside in the default system tenant) need only meet the following criteria.

• The administrator must have administrative access on the tenant to which the user belongs.

• The administrator must have the OnBehalfOf (21) permission.

• The administrator must have permission for any operation he or she plans to execute. For example, the administrator must have the DeleteDataSource permission to be able to delete a data source on behalf of a user.

For a summary list of supported on-behalf-of API operations and specific permissions for each, see On-behalf-of API operations on page 1282.
See also
User provisioning on page 107

On-behalf-of API operations
This reference provides a list of operations that can be carried out by an administrator on behalf of a user. As shown, an administrator can execute these operations by appending the user query parameter to the request. (See also User provisioning on page 107.)

• General data source operations on page 1282
• OData operations on page 1283
• Group data source operations on page 1284
• Permissions operations on page 1285
• Data source sharing operations on page 1286
• OAuth application object operations for Google Analytics connectivity on page 1287
• OAuth profile object operations for Google Analytics connectivity on page 1288

General data source operations
Operation: Create a data source or group data source
Request: POST https://<myserver>:<port>/api/mgmt/datasources?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the CreateDataSource (1) permission.

Operation: Retrieve a list of data sources
Request: GET https://<myserver>:<port>/api/mgmt/datasources?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Retrieve the details for a data source
Request: GET https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Update the options and values for a data source
Request: PUT https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Delete a data source
Request: DELETE https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the DeleteDataSource (4) permission.

Operation: Test a connection to a data source
Request: POST
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/test?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, the ViewDataSource (2) permission, and at least one query permission such as UseDataSourceWithJDBC (5), UseDataSourceWithODBC (6) or UseDataSourceWithOData (7).

Operation: Refresh the cached object mapping of a data source
Request: POST
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/map?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

OData operations

Operation: Create or refresh a data source OData model
Request: POST
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/model?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Check status of the OData model refresh
Request: GET
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/model?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Retrieve a list of available schemas
Request: GET
https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Retrieve table names
Request:

• For data stores that support schemas
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables

• For data stores that do not support schemas
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/-/tables
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: **Retrieve table information**

Request:
- For data stores that support schemas
  
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables/<tableName>?user=<userName>

- For data stores that do not support schemas
  
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/-/tables/<tableName>?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: **Retrieve column information for a table**

Request:
- For data stores that support schemas
  
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables/<tableName>/columns?user=<userName>

- For data stores that do not support schemas
  
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/-/tables/<tableName>/columns?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: **Retrieve primary keys for a table**

Request:
- For data stores that support schemas
  
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables/<tableName>/primarykeys?user=<userName>

- For data stores that do not support schemas
  
  GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/-/tables/<tableName>/primarykeys?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

**Group data source operations**

Operation: **Create a data source or group data source**
Request: POST https://<myserver>:<port>/api/mgmt/datasources?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the CreateDataSource (1) permission.

Operation: Retrieve the members of a group data source

Request: GET
https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Add member data sources to a group data source

Request: POST
https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Update members of a group data source

Request: POST https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Delete a member data source from a group data source

Request: DELETE
https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members/{memberDatasourceId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Permissions operations

Operation: Retrieve permissions on a data source

Request: GET
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/permissions?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Update permissions on data sources

Request: PUT
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/permissions?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Retrieve the user's permissions

Request: GET https://<myserver>:<port>/api/mgmt/permissions?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, and the OnBehalfOf (21) permission.

Data source sharing operations

Operation: Retrieve users with whom data source is being shared
Request: GET
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Share data source with a user or users
Request: POST
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Stop sharing the data source with users
Request: DELETE
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the DeleteDataSource (4) permission.

Operation: Retrieve the data source permissions for a user with whom the data source is being shared
Request: GET
https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}/sharedUsers/{userId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Update the data source permissions for a user with whom the data source is being shared
Request: PUT
https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}/sharedUsers/{userId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Stop sharing the data source with a user
Request:
https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}/sharedUsers/{userId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Retrieve tenants with which the data source is being shared
Request: GET
https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}/sharedTenants?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Share data source with a tenant or tenants

Request: POST
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Stop sharing a data source with tenants

Request: DELETE
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Retrieve the data source permissions for a tenant with which the data source is being shared

Request: GET
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants/{tenantId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Update the data source permissions for a tenant with which the data source is being shared

Request: PUT
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants/{tenantId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Stop sharing the data source with a tenant

Request: DELETE
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants/{tenantId}?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

OAuth application object operations for Google Analytics connectivity

Operation: Retrieve OAuth applications

Request: GET
https://<myserver>:<port>/api/mgmt/oauthapps?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the OAuth (28) permission.

Operation: Create an OAuth application object

Request: POST
https://<myserver>:<port>/api/mgmt/oauthapps?user=<userName>

Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the OAuth (28) permission.
Chapter 10: Hybrid Data Pipeline API Reference

Operation: Retrieve an OAuth application object
Request: GET https://<myserver>:<port>/api/mgmt/oauthapps/{id}?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the OAuth (28) permission.

Operation: Update an OAuth application object
Request: PUT https://<myserver>:<port>/api/mgmt/oauthapps/<id>?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the OAuth (28) permission.

Operation: Delete an OAuth application object
Request: DELETE https://<myserver>:<port>/api/mgmt/oauthapps/{id}?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the OAuth (28) permission.

OAuth profile object operations for Google Analytics connectivity

Operation: Retrieve OAuth profiles
Request: GET https://<myserver>:<port>/api/mgmt/oauthprofiles?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Create an OAuth profile
Request: POST https://<myserver>:<port>/api/mgmt/oauthprofiles?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the CreateDataSource (1) permission.

Operation: Retrieve an OAuth profile
Request: GET https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

Operation: Update an OAuth profile
Request: PUT https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Delete an OAuth profile
Request: DELETE https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}?user=<userName>
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

Operation: Retrieve statistics for an OAuth profile
**Request:** GET
https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}/stats?user=<userName>

**Permissions:** The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ViewDataSource (2) permission.

---

**Get data stores**

**Purpose**
Retrieves a list of supported backend data stores and their options.

**URL**
https://<myserver>:<port>/api/mgmt/datastores

**Method**
GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Definition**
The response takes the following format. The properties of the response are described in the table that follows.

```
{
    "datastores": [  
        {
            "id": datastore_id,
            "name": "datastore_name",
            "isBeta": boolean,
            "isGroup": boolean,
            "authorized": boolean,
            "connectionType": {connection_type_details}
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The integer ID of the data store</td>
<td>The data store ID</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the data store</td>
<td>The name of the data store</td>
</tr>
<tr>
<td>&quot;isBeta&quot;</td>
<td>Indicates whether the data store is beta or GA</td>
<td>true</td>
</tr>
</tbody>
</table>

- If true, the data store is a beta data store.
- If false, the data store is GA.
<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;isGroup&quot;</td>
<td>Indicates whether the data store is the group data store (as opposed to a specific back end data store such as Oracle)</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>The group data store enables the creation of group data sources. A group data source is comprised of multiple member data sources that connect to one or more back end data stores such as Salesforce or SQL Server.</td>
<td>If true, the data store is the group data store. If false, the data store is not the group data store.</td>
</tr>
<tr>
<td></td>
<td>The group data store is named DataSource Group, and its ID is 56.</td>
<td></td>
</tr>
<tr>
<td>&quot;authorized&quot;</td>
<td>Indicates whether the user making the request can create a data source on the data store</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>If true, the user making the request is authorized to create a data source for this data store. If false, the user making the request is not authorized to create a data source for this data store.</td>
<td></td>
</tr>
<tr>
<td>&quot;connectionType&quot;</td>
<td>Provides details about a supported data store, including the options that can be specified when creating a data source on the data store.</td>
<td>A valid connectionType object. See connectionType-details Object on page 1293 for more information.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

```json
{
    "dataStores": [
        {
            "id": 1,
            "name": "Salesforce",
            "isBeta": false,
            "isGroup": false,
            "authorized": true,
            "connectionType": [  
                {
                    "name": "Cloud",
                    "category": [  
                        {
                            "name": "General",
                            "options": [  
                                {
                                    "id": "Name",
                                    "displayName": "Data Source Name",
                                    "documentation": "A name you provide to uniquely identify this Data Source.",
                                    "required": true,
                                    "maxLength": 128,
                                    "type": "string"
                                },
                                ...
                            ]
                        }  
                    ],
                    "id": "SecurityToken",
                    "displayName": "Security Token",
                    "category": [  
                        {
                            "name": "General",
                            "options": [  
                                {
                                    "id": "Name",
                                    "displayName": "Data Source Name",
                                    "documentation": "A name you provide to uniquely identify this Data Source."
                                },
                                ...
                            ]
                        }  
                    ],
                    "category": [  
                        {
                            "name": "General",
                            "options": [  
                                {
                                    "id": "Name",
                                    "displayName": "Data Source Name",
                                    "documentation": "A name you provide to uniquely identify this Data Source."
                                },
                                ...
                            ]
                        }  
                    ],
                    "category": [  
                        {
                            "name": "General",
                            "options": [  
                                {
                                    "id": "Name",
                                    "displayName": "Data Source Name",
                                    "documentation": "A name you provide to uniquely identify this Data Source."
                                },
                                ...
                            ]
                        }  
                    ],
                    "category": [  
                        {
                            "name": "General",
                            "options": [  
                                {
                                    "id": "Name",
                                    "displayName": "Data Source Name",
                                    "documentation": "A name you provide to uniquely identify this Data Source."
                                },
                                ...
                            ]
                        }  
                    ],
                    "category": [  
                        {
                            "name": "General",
                            "options": [  
                                {
                                    "id": "Name",
                                    "displayName": "Data Source Name",
                                    "documentation": "A name you provide to uniquely identify this Data Source."
                                },
                                ...
                            ]
                        }  
                    ],
                    "category": [  
                        {
                            "name": "General",
                            "options": [  
                                {
                                    "id": "Name",
                                    "displayName": "Data Source Name",
                                    "documentation": "A name you provide to uniquely identify this Data Source."
                                },
                                ...
                            ]
                        }  
                    ]
                },
```

---

**Chapter 10: Hybrid Data Pipeline API reference**

**Progress DataDirect Hybrid Data Pipeline: User's Guide: Version 4.6**

---

1290
"documentation": "The security token is required to log in to Salesforce from an untrusted network. ... A new token will be sent by e-mail.", "type": "string"
],
...
{
  "name": "Advanced",
  "options": [
  {
    "id": "StmtCallLimit",
    "displayName": "Web Service Call Limit",
    "documentation": "The maximum number of Web service calls allowed to Salesforce for a single SQL statement or metadata query.",
    "minInclusive": 0,
    "maxInclusive": 2000000000,
    "type": "integer",
    "default": "0"
  },
  ...
  {
    "id": "HDFMetadataExposedSchemas",
    "displayName": "Metadata Exposed Schemas",
    "documentation": "Defines the schemas to be allowed in the metadata queries.",
    "type": "string"
  }
  }
}]
},
...
{
  "id": 43,
  "name": "Oracle",
  "isBeta": false,
  "isGroup": false,
  "authorized": true,
  "connectionType": [
  {
    "name": "Hybrid",
    "category": [
    {
      "name": "General",
      "options": [
      {
        "id": "Name",
        "displayName": "Data Source Name",
        "documentation": "A name you provide to uniquely identify this Data Source.",
        "required": true,
        "maxLength": 128,
        "type": "string"
      },
      ...
      {
        "id": "TNSServerName",
        "displayName": "TNS Server Name",
        "documentation": "The Oracle net service name that is used to reference the connection information

in a tnsnames.ora file.
"type": "string"
}
  },
...  
  
  "name": "Advanced",
  "options": [
    
    
    
    "id": "AlternateServers",
    "displayName": "Alternate Servers",
    "documentation": "The server name (servername1, servername2, and so on) is required for each 
    ... default port number of 1521 is used. For 
    more information, see the Help.",
    "type": "string"
  ],
  ...
  
  "id": "HDPMetadataExposedSchemas",
  "displayName": "Metadata Exposed Schemas",
  "documentation": "Defines the schemas to be allowed 
  in the metadata queries.",
  "type": "string"
}
]

Sample Server Failure Response

{
  "error": {
    "code": "222206007",
    "message": {
      "lang": "en-US",
      "value": "Invalid user ID or password"
    }
  }
}

Authentication

Basic Authentication using Login ID and Password.

Authorization

The user must have the MgmtAPI (11) permission.

Related topics

- Get data stores on page 1289
- connectionType-details Object on page 1293
connectionType-details Object

Purpose
Describes the connectionType details information for a data store. The connectionType-details object defines the list of options that can be specified when creating a data source for the data store type. The options in the list can be grouped into categories. A category is a group of options that are related in some way. For example, a Security category may group together options such as Encryption Method or TLS/SSL Protocol Version.

The dataStores value is an array of `datastore-info` objects for all of the available data stores. Each successful response has only one dataStores element.

Syntax

```
{  
  connectionType-details
  {  
    "name": <connectionType-name>,
    "category": [  
      {category-definition  
        {  
          "name": <category-name>,
          "options": [{option-definition}]
        }
      }
    ]
  }
}
```

collectionType-details Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The connectionType name. Currently, the name must be either Cloud or Hybrid.</td>
<td>Yes</td>
</tr>
<tr>
<td>&quot;category&quot;</td>
<td>An array of one or more category definition objects. A category is a group of options for a data store that are loosely related to each other. The Hybrid Data Pipeline web interface displays each category of options on a separate tab in its data source dialogs. A category-definition object has the format:</td>
<td>Yes</td>
</tr>
</tbody>
</table>

```
{  
  "name": <category-name>,
  "options": [{option-definition}]
}
```
Related topics

- Get data stores on page 1289
- category-definition Object on page 1294
- option-definition Object on page 1295
- choice-definition Object on page 1297

category-definition Object

Purpose

Describes the category-definition details for a data store connectionType object.

The category-definition contains a list of options that can be set on a DataSource based on this data store. A data store can have one or more categories of options.

Syntax

```
{  
    "category-definition": {  
        "name": <category-name>,  
        "options": [{option-definition}]
    }
}
```

category-definition Object Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Valid Values</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The category name. The category can be anything the connectivity service defines, such as General, Advanced, Security, and OData.</td>
<td>Yes</td>
</tr>
<tr>
<td>options</td>
<td>An array of one or more option-definition objects. The option definition defines the option id, display name, data type, and other information to describe the option. A option-definition object has the format:</td>
<td>No</td>
</tr>
</tbody>
</table>

```
{  
    "id": <option-id>,  
    "displayName": <display-name>,  
    "documentation": "<help-text>",  
    "required": (true | false),  
    "type": <option-data-type>,  
    "default": <default-value>,  
    "choices": [{choice-definition}]
}
```

See option-definition Object on page 1295 for more information.
Related topics

- Get data stores on page 1289
- connectionType-details Object on page 1293
- option-definition Object on page 1295
- choice-definition Object on page 1297

option-definition Object

Purpose
Describes properties of an option that can be set on a data source based on this data store type.

Syntax

```json
{
  "option-definition": {
    "id": <option-id>,
    "displayName": <display-name>,
    "documentation": "<help-text>",
    "required": (true | false),
    "type": <option-data-type>,
    "default": <default-value>,
    "choices": [{choice-definition}]
  }
}
```
## option-definition Object Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Valid Values</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The ID of the option. The ID is used to identify the option when setting or fetching the option value when referencing a data source.</td>
<td>Yes</td>
</tr>
<tr>
<td>displayName</td>
<td>A user-friendly name for the option that can be used in UI displays and other cases where the option is exposed to the end user.</td>
<td>Yes</td>
</tr>
<tr>
<td>documentation</td>
<td>A brief description of the option that can be displayed as help text for the end user.</td>
<td>No</td>
</tr>
<tr>
<td>required</td>
<td>If set to true, a value must be set for the option when creating or updating a data source. If set to false or not specified, the option is not required.</td>
<td>No</td>
</tr>
<tr>
<td>type</td>
<td>The data type of the option. Currently, the following data types are supported:&lt;br&gt;• boolean&lt;br&gt;• string&lt;br&gt;• integer&lt;br&gt;• password. An option with a data type of password indicates that the value for this option contains sensitive information such as a password or security token. Applications should provide the appropriate precautions when displaying a value with the password data type.</td>
<td>Yes</td>
</tr>
<tr>
<td>default</td>
<td>The value used for the option if a value is not specified for the data source.</td>
<td>No</td>
</tr>
<tr>
<td>choices</td>
<td>An array of choice-definition objects that define the set of valid values for the option. A string option may be restricted to a set of one or more valid string values. A choice-definition object has the format:&lt;br&gt;<code>{&lt;br&gt;  &quot;id&quot;: &lt;choice-value&gt;,&lt;br&gt;  &quot;name&quot;: &lt;display-value&gt;  }&lt;br&gt;</code>&lt;br&gt;See choice-definition Object on page 1297 for more information.</td>
<td>No</td>
</tr>
</tbody>
</table>

### Related topics
- [Get data stores](#) on page 1289
- [connectionType-details Object](#) on page 1293
- [category-definition Object](#) on page 1294
- [choice-definition Object](#) on page 1297
choice-definition Object

Purpose
Describes the choice-definition details information for the options object of an option-definition object.

Syntax

```json
{choice-definition
    {
        "id": <choice-value>,
        "name": <display-value>
    }
}
```

choice-definition Object Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Valid Values</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The value to be used when setting the data source option if this choice is selected.</td>
<td>Yes</td>
</tr>
<tr>
<td>name</td>
<td>A user-friendly version of the option value that can be used to display to the user.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Related topics

- Get data stores on page 1289
- connectionType-details Object on page 1293
- category-definition Object on page 1294
- option-definition Object on page 1295

Get options for a data store

Purpose
Retrieves information and options for the specified data store. The options available on the data source are returned in the connectionType object. These options may be specified when creating a data source on the specified data store.

URL

https://<myserver>:<port>/api/mgmt/datastores/{datastoreId}

Method

GET
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

{datastoreId} is the integer ID of the data store. This data store ID is used to identify the data store in data source references.

Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
   "id": datastore_id,
   "name": "datastore_name",
   "isBeta": boolean,
   "isGroup": boolean,
   "authorized": boolean,
   "connectionType": {connection_type_details}
}
```

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The integer ID of the data store</td>
<td>The data store ID</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the data store</td>
<td>The name of the data store</td>
</tr>
<tr>
<td>&quot;isBeta&quot;</td>
<td>Indicates whether the data store is beta or GA</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>If true, the data store is a beta data store.</td>
<td>If true, the data store is beta data store.</td>
</tr>
<tr>
<td></td>
<td>If false, the data store is GA.</td>
<td>If false, the data store is GA.</td>
</tr>
<tr>
<td>&quot;isGroup&quot;</td>
<td>Indicates whether the data store is the group data store (as opposed to a specific back end data store such as Oracle)</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>The group data store enables the creation of group data sources. A group data source is comprised of multiple member data sources that connect to one or more back end data stores such as Salesforce or SQL Server.</td>
<td>If true, the data store is the group data store.</td>
</tr>
<tr>
<td></td>
<td>The group data store is named DataSource Group, and its ID is 56.</td>
<td>If false, the data store is not the group data store.</td>
</tr>
</tbody>
</table>

### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;authorized&quot;</td>
<td>Indicates whether the user making the request can create a data source on the data store</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>If true, the user making the request is authorized to create a data source for this data store.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If false, the user making the request is not authorized to create a data source for this data store.</td>
<td></td>
</tr>
<tr>
<td>&quot;connectionType&quot;</td>
<td>Provides details about a supported data store, including the options that can be specified when creating a data source on the data store.</td>
<td>A valid connectionType object. See connectionType-details Object on page 1293 for more information.</td>
</tr>
</tbody>
</table>

### Sample Server Success Response

```json
{
    "id": 1,
    "name": "Salesforce",
    "isBeta": false,
    "isGroup": false,
    "authorized": true,
    "connectionType": [
        {
            "name": "Cloud",
            "category": [
                {
                    "name": "General",
                    "options": [
                        {
                            "id": "Name",
                            "displayName": "Data Source Name",
                            "documentation": "A name you provide to uniquely identify this Data Source.",
                            "required": true,
                            "maxLength": 128,
                            "type": "string"
                        },
                        ...
                        {
                            "id": "SecurityToken",
                            "displayName": "Security Token",
                            "documentation": "The security token is required to log in to Salesforce from an untrusted network. Salesforce... A new token will be sent by e-mail.",
                            "type": "string"
                        }
                    ]
                },
                ...
            ]
        },
        ...
        {
            "name": "Advanced",
            "options": [
                {
                    "id": "StmtCallLimit",
                    "displayName": "Web Service Call Limit",
                    "documentation": "The maximum number of Web service calls allowed to Salesforce for a single SQL statement or metadata query.",
                    "minInclusive": 0,
                    "maxInclusive": 2000000000,
                }
            ]
        }
    ]
}
```
Sample Server Failure Response

```
{
  "error": {
    "code": "222207015",
    "message": {
      "lang": "en-US",
      "value": "Invalid DataStore ID: 88"
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) permission.

**Create a data source**

**Purpose**

Creates a data source or group data source. The user who creates the data source is the owner of the data source. When an administrator creates a data source on behalf of a user, the user identified with the user query parameter is the owner of the data source.

**Note:** A group data source is a data source that is comprised of member data sources. The creation and configuration of group data sources allows a single OData endpoint to be configured for multiple member data sources. See Configuring data sources for OData connectivity and working with data source groups on page 622 for details.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.
**Note:** Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user.

**URL**

https://<myserver>:<port>/api/mgmt/datasources

**Method**

POST

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Request Payload Definition**

The request takes the following format. The properties of the request are described in the table that follows.

**Note:** The values for "dataStore" and "connectionType" cannot be changed once the data source is created. A new data source must be created if any of these values need to be changed.

```
{
   "name": "datasource_name",
   "dataStore": datastore_id,
   "connectionType": "connection_type",
   "description": "datasource_description",
   "options": {
      "option1": "option1_value",
      "option2": "option2_value",
      ...
   },
   "permissions": [integer, integer, ...],
   "members": ["datasource1", "datasource2", ...]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the data source. This name is passed as a database parameter when establishing a connection to the data source with the ODBC driver, the JDBC driver, or the OData API.</td>
<td>Required</td>
<td>The first character of the name must be a letter, and the name can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
</tbody>
</table>
### Parameter | Description | Usage | Valid Values
--- | --- | --- | ---
"dataStore" | The ID of the data store on which the data source is being created. The data store defines the options that can be specified when creating the data source. Group data sources must be created on the Hybrid Data Pipeline group data store. A group data source is comprised of multiple member data sources that connect to one or more back end data stores such as Salesforce or SQL Server. | Required | The integer ID of the data store. If you are creating a group data source, this property must be set to 56 to specify the DataSource Group data store. Data store IDs can be obtained with the Get data stores call. |
"connectionType" | Specifies whether the data source is a cloud, hybrid, or group data source | Required | "Cloud" | "Hybrid" | Group |
"description" | A description of the data source | Optional | A description of the data source provided by the user who created the data source |
"options" | The list of option names and values to be set on the data source. The list of allowed options depends on the data store. Data store options can be retrieved with the Get options for a data store on page 1297 call. | Required | A comma separated list of options and their values. The content of the options object is zero or more sets of option names and values. If no options are to be set on the data source, specify an empty object {}. |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated explicitly with the data source. Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user. Any permissions specified for this data source will override the permissions for the user or the user's role that own this data source. You must specify the exact set of permissions that you want to set for this data source as no permissions are inherited from the user or user's role if permissions are specified on a data source. Permissions set on a group data source override permissions set on any of its member data sources.</td>
<td>Optional</td>
<td>A comma separated list of permission IDs See Data source permissions on page 1321 for supported permissions.</td>
</tr>
<tr>
<td>&quot;members&quot;</td>
<td>The members object can be used to assign member data sources to a group data source. A member data source cannot itself be a group data source. Member data sources can be assigned when a group data source is being created or added after the group data source has been created.</td>
<td>Optional</td>
<td>The members object includes an &quot;id&quot; property and an &quot;entityPrefix&quot; property. The &quot;id&quot; specifies the ID of a member data source. The member data source cannot itself be a group data source. The &quot;entityPrefix&quot; is a user-defined prefix associated with a specific data source to resolve naming conflicts. The prefix must be 1 to 64 characters in length and must be unique.</td>
</tr>
</tbody>
</table>

**Example 1: Request Payload**

In this example, a standard user creates a data source on a Salesforce data store.

```
POST https://Server03:8443/api/mgmt/datasources

{
    "name": "SF2",
    "dataStore": "1",
    "connectionType": "Cloud",
    "description": "Test Salesforce access",
    "options": {
        "Database": "Accounting",
        "User": "mySForceUserld",
        "Password": "mySForcePassword",
        "SecurityToken": "mySecurityToken",
        "StmtCallLimit": "60"
    }
}
```
**Success Response**

Status code: 201  
Successful response

```
{
  "id": "5039",
  "name": "SF2",
  "dataStore": "1",
  "connectionType": "Cloud",
  "description": "Test Salesforce access",
  "options": {
    "Database": "Accounting",
    "User": "mySForceUser1d",
    "Password": "mySForcePassword",
    "SecurityToken": "mySecurityToken",
    "StmtCallLimit": "60"
  }
}
```

**Example 2: Request Payload**

In this example, an administrator creates a data source with permissions on behalf of a user. The user’s access to the data store is restricted by the permissions.

https://Server03:8443/api/mgmt/datasources?user=user11

```
{
  "name": "SF2",
  "dataStore": "1",
  "connectionType": "Cloud",
  "description": "Test Salesforce access",
  "options": {
    "Database": "Accounting",
    "User": "mySForceUser1d",
    "Password": "mySForcePassword",
    "SecurityToken": "mySecurityToken",
    "StmtCallLimit": "60"
  },
  "permissions": [
    1,
    2,
    3,
    4,
    5
  ]
}
```

**Success Response**

Status code: 201  
Successful response

```
{
  "id": "6444",
  "name": "SF2",
  "dataStore": "1",
  "connectionType": "Cloud",
  "description": "Test Salesforce access",
  "options": {
    "Database": "Accounting",
    "User": "mySForceUser1d",
    "Password": "mySForcePassword",
    "SecurityToken": "mySecurityToken",
    "StmtCallLimit": "60"
  }
}
```
Example 3: Request Payload

In this example, a standard user creates a group data source for testing OData access.

https://Server03:8443/api/mgmt/datasources

{
  "name": "OData_Group",
  "dataStore": "56",
  "connectionType": "Group",
  "description": "Test OData connectivity",
  "options": {
    "Name": "OData_Group",
    "Description": "Test OData connectivity",
    "ODataVersion": "4",
    "MaximumEntityNameLength": "64"
  },
  "members": [
    {
      "id": 3,
      "entityPrefix": "fin"
    },
    {
      "id": 6,
      "entityPrefix": "mkt"
    }
  ]
}

Sample Success Responses

Status code: 201
Successful response

{
  "id": "7255",
  "name": "OData_Group",
  "dataStore": "56",
  "connectionType": "Group",
  "description": "Test OData connectivity",
  "options": {
    "Name": "OData_Group",
    "Description": "Test OData connectivity",
    "ODataVersion": "Version 4",
    "MaximumEntityNameLength": 64,
    "ODataDataSourceCaching": 1
  },
  "members": [
    {
      "id": 3,
      "entityPrefix": "fin"
    },
    {
      "id": 6,
      "entityPrefix": "mkt"
    }
  ]
}
Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and CreateDataSource (1) permissions.

See also
Create or refresh a data source OData model on page 1328
Configuring data sources for OData connectivity and working with data source groups on page 622

Get data sources

Purpose
Retrieves a list of data sources with details for each including the data source ID. The data source ID can be used to retrieve further details for each data source, or carry out other operations on a given data source.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources

Filter by a query parameter
The list of data sources can also be filtered by member, type, and isOData filters. Use a semicolon to separate query parameters when filtering by more than one parameter.

- URL with filter by the member parameter.
  https://<myserver>:<port>/api/mgmt/datasources?member=<member_datasource_id>

- URL with filter by the type parameter.
  https://<myserver>:<port>/api/mgmt/datasources?type=<datasource_type>

- URL with filter by the isOData parameter.
  https://<myserver>:<port>/api/mgmt/datasources?isOData=<boolean>

The following table describes query parameters that can be used to filter the list of data sources.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;member&quot;</td>
<td>Allows the list to be filtered to include only data source groups for which the specified data source is a member</td>
<td>A valid member data source ID</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>&quot;type&quot;</td>
<td>Allows the list to be filtered based on group status</td>
<td>all</td>
</tr>
<tr>
<td></td>
<td>If set to all, all data sources are returned whether or not they are group data sources.</td>
<td>If set to simple, only data sources that are not group data sources are returned. However, the list will include data sources that are members of a data source group.</td>
</tr>
<tr>
<td>&quot;isOData&quot;</td>
<td>Allows the list to be filtered by data sources that have been enabled for OData</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>If true, only data sources that are OData enabled are returned.</td>
<td>If false, only non-OData data sources are returned.</td>
</tr>
</tbody>
</table>

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Response Definition**

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
   "dataSources": [
      {
         "id": "datasource_id",
         "name": "datasource_name",
         "dataStore": datastore_id,
         "isGroup": boolean,
         "description": "datasource_description",
         "sharedByAnotherUser": boolean,
         "sharedWithAnotherUser": boolean,
         "permissions": [integer, integer, ...]
      }
   ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the data source</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the data source. This name is passed as a database parameter when establishing a connection to the data source with the ODBC driver, the JDBC driver, or the OData API.</td>
<td>The first character of the name must be a letter, and the name can contain only alphanumeric characters, underscores and dashes.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store on which the data source is being created. The data store defines the options that can be specified when creating the data source. Group data sources must be created on the Hybrid Data Pipeline group data store. A group data source is comprised of multiple member data sources that connect to one or more backend data stores such as Salesforce or SQL Server.</td>
<td>The integer ID of the data store Data store IDs can be obtained with the Get data stores call.</td>
</tr>
</tbody>
</table>
| "isGroup"     | Indicates whether the data source is a group data source. A group data source is comprised of member data sources. | true | false  
If true, the data source is a group data source.  
If false, the data source is not a group data source. |
<p>| &quot;description&quot; | A description of the data source                                             | A description of the data source provided by the user who created the data source |
| &quot;sharedByAnotherUser&quot; | Indicates whether the data source is being shared by another user. Provided only when the data source is shared by another user. | true when the data source is being shared by another user. |</p>
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;sharedWithAnotherUser&quot;</td>
<td>Indicates whether the data source is being shared with another user.</td>
<td>true when the data source is being shared with another user.</td>
</tr>
<tr>
<td></td>
<td>Provided only when the data source is shared with another user.</td>
<td></td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated explicitly with the data source.</td>
<td>A comma separated list of permission IDs supported on page 1321 for permissions.</td>
</tr>
<tr>
<td></td>
<td>Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Any permissions specified for this data source will override the permissions for the user or the user's role that own this data source. You must specify the exact set of permissions that you want to set for this data source as no permissions are inherited from the user or user's role if permissions are specified on a data source.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Permissions set on a group data source override permissions set on any of its member data sources.</td>
<td></td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

**Note:** The response will not return settings for optional properties that were not set in a previous POST or PUT request.

```json
{
  "dataSources": [
    {
      "id": "18",
      "name": "Oracle_Test",
      "dataStore": 43,
      "isGroup": false,
      "description": "Oracle data source on test schema",
      "permissions": [
        1,
        2,
        3,
        4,
        5
      ]
    },
    ...
  ]
}
```
Sample Server Failure Response

```json
{
    "error": {
        "code": 222207004,
        "message": {
            "lang": "en-US",
            "value": "There is no DataSource with that id: 1234."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

**Get data source details**

**Purpose**

Retrieves the details for a specified data source. The details include the ID and name of the data source, the options specified for the data source, and other information, such as whether the data source is a member of a data source group.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Response Definition**

The response takes the following format. The properties of the response are described in the table that follows.
**Note:** The `sharedByAnotherUser` and `sharedWithAnotherUser` properties will only be included in the response when the `?details=true` parameter is appended to the query and the actual value of either property is `true`.

```
{
    "id": "datasource_id",
    "name": "datasource_name",
    "dataStore": datastore_id,
    "connectionType": "connection_type",
    "description": "datasource_description",
    "sharedByAnotherUser": boolean,
    "sharedWithAnotherUser": boolean,
    "options": {
        "option1": "option1_value",
        "option2": "option2_value",
        ...
    },
    "permissions": [integer, integer, ...],
    "members": ["datasource1", "datasource2", ...]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the data source</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the data source. This name is passed as a database parameter when establishing a connection to the data source with the ODBC driver, the JDBC driver, or the OData API.</td>
<td>The first character of the name must be a letter, and the name can contain only alphanumeric, underscores, and dashes.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store on which the data source is being created. The data store defines the options that can be specified when creating the data source.</td>
<td>The integer ID of the data store</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For a group data source, this property is set to 56 to specify the <code>DataSource Group</code> data store.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data store IDs can be obtained with the <code>Get data stores</code> call.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Group data sources must be created on the Hybrid Data Pipeline group data store. A group data source is comprised of multiple member data sources that connect to one or more back end data stores such as Salesforce or SQL Server.</td>
</tr>
</tbody>
</table>

**Management API**
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
</table>
| "connectionType"   | Specifies whether the data source is a cloud, hybrid, or group data source   | "Cloud" | "Hybrid" | Group
If set to "Cloud", the data source is accessible from the public WAN.
If set to "Hybrid" the data source is a hybrid data source. Depending on how it is configured, a hybrid data source can connect to either a public WAN data source or to a data source behind a firewall using the On-Premises Connector to create a cloud-only data source.
If set to Group, the data source is a group data source. A group data source must be created on the DataSource Group data store by setting the "dataStore" property to 56.

<table>
<thead>
<tr>
<th>&quot;description&quot;</th>
<th>A description of the data source</th>
<th>A description of the data source provided by the user who created the data source</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;sharedByAnotherUser&quot;</td>
<td>Indicates whether the data source is being shared by another user. Provided only when the ?details=true parameter is appended to the query and the data source is being shared by another user.</td>
<td>true when the data source is being shared by another user.</td>
</tr>
</tbody>
</table>

| "sharedWithAnotherUser" | Indicates whether the data source is being shared with another user. Provided only when the ?details=true parameter is appended to the query and the data source is being shared with another user. | true when the data source is being shared with another user. |

<p>| &quot;options&quot; | The list of option names and values to be set on the data source. The list of allowed options depends on the data store. Data store options can be retrieved with the Get options for a data store on page 1297 call. | A comma separated list of options and their values. The content of the options object is zero or more sets of option names and values. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated explicitly with the data source. Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user. Any permissions specified for this data source will override the permissions for the user or the user's role that own this data source. You must specify the exact set of permissions that you want to set for this data source as no permissions are inherited from the user or user's role if permissions are specified on a data source. Permissions set on a group data source override permissions set on any of its member data sources.</td>
<td>A comma separated list of permission IDs. See Data source permissions on page 1321 for supported permissions.</td>
</tr>
<tr>
<td>&quot;members&quot;</td>
<td>The members object can be used to assign member data sources to a group data source. Member data sources can be assigned when a group data source is being created or added after the group data source has been created.</td>
<td>The members object includes an &quot;id&quot; property and an &quot;entityPrefix&quot; property. The &quot;id&quot; specifies the ID of a member data source. The member data source cannot itself be a group data source. The &quot;entityPrefix&quot; is a user-defined prefix associated with a specific data source to resolve naming conflicts. The prefix must be 1 to 64 characters in length and must be unique.</td>
</tr>
</tbody>
</table>

Sample Server Response

**Note:** The response will not return settings for optional properties that were not set in a previous POST or PUT request.

**Example 1**

```json
{
  "id": "5039",
  "name": "SF2",
  "dataStore": 1,
  "connectionType": "Cloud",
  "description": "Test",
  "options": {
    "User": "mysfusername",
    "Password": "mysfpassword",
    "SecurityToken": "mysecuritytoken",
    "EnableBulkLoad": "true",
```
Example 2

The following server response is for a group data source. As shown here, a "members" array is returned for group data sources.

```
Status code: 200
Successful response

{
  "id": "5051",
  "name": "OData_Group",
  "dataStore": "56",
  "connectionType": "Group",
  "description": "Test OData connectivity",
  "options": {
    "Name": "OData_Group",
    "Description": "Test OData connectivity",
    "ODataVersion": "4",
    "MaximumEntityNameLength": "64"
  },
  "permissions": [
    1,
    2,
    3,
    4,
    5
  ],
  "members": [
    {
      "id": 3,
      "entityPrefix": "fin"
    },
    {
      "id": 6,
      "entityPrefix": "mkt"
    }
  ]
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.
Update a data source

Purpose
Updates the details of an existing data source.
When using OData, you must refresh the OData data model after updating a data source.

**Note:** The "id", "dataStore", and "connectionType" properties of a data source cannot be changed. These properties can be passed in the payload to update the data source, but they must match the current values set in the data source.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**Note:** Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user.

**URL**
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}

**Method**
PUT

**URL Parameters**
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Request Payload Definition**
The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "name": "datasource_name",
    "dataStore": "dataStore_id",
    "connectionType": "connection_type",
    "description": "datasource_description",
    "options": {
        "option1": "option1_value",
        "option2": "option2_value",
        ...
    }
}
```
```
"permissions": [integer, integer, ...],
"members": ["datasource1", "datasource2", ...]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the data source. This name is passed as a database parameter when establishing a connection to the data source with the ODBC driver, the JDBC driver, or the OData API.</td>
<td>Required</td>
<td>The first character of the name must be a letter, and the name can contain only alphanumeric characters, underscores, and dashes.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store on which the data source is being created. The data store defines the options that can be specified when creating the data source. Group data sources must be created on the Hybrid Data Pipeline group data store. A group data source is comprised of multiple member data sources that connect to one or more back end data stores such as Salesforce or SQL Server.</td>
<td>Required</td>
<td>The integer ID of the data store. For a group data source, this property is set to 56 to specify the DataSource Group data store. Data store IDs can be obtained with the Get data stores call.</td>
</tr>
<tr>
<td>&quot;connectionType&quot;</td>
<td>Specifies whether the data source is a cloud, hybrid, or group data source</td>
<td>Required</td>
<td>&quot;Cloud&quot;</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the data source</td>
<td>Optional</td>
<td>A description of the data source provided by the user who created the data source</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;options&quot;</td>
<td>The list of option names and values to be set on the data source. The list of allowed options depends on the data store. Data store options can be retrieved with the Get options for a data store on page 1297 call.</td>
<td>Required</td>
<td>A comma separated list of options and their values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The content of the options object is zero or more sets of option names and values.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated explicitly with the data source. Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user. Any permissions specified for this data source will override the permissions for the user or the user's role that own this data source. You must specify the exact set of permissions that you want to set for this data source as no permissions are inherited from the user or user's role if permissions are specified on a data source. Permissions set on a group data source override permissions set on any of its member data sources.</td>
<td>Optional</td>
<td>A comma separated list of permission IDs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>See Data source permissions on page 1321 for supported permissions.</td>
</tr>
<tr>
<td>&quot;members&quot;</td>
<td>The members object can be used to assign member data sources to a group data source. Member data sources can be assigned when a group data source is being created or added after the group data source has been created.</td>
<td>Optional</td>
<td>The members object includes an &quot;id&quot; property and an &quot;entityPrefix&quot; property.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The &quot;id&quot; specifies the ID of a member data source. The member data source cannot itself be a group data source.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The &quot;entityPrefix&quot; is a user-defined prefix associated with a specific data source to resolve naming conflicts. The prefix must be 1 to 64 characters in length and must be unique.</td>
</tr>
</tbody>
</table>

### Sample Request Payload

Note: Optional properties not included in the payload request will be removed from the object.

```json
{
    "name": "SF2",
}"
"dataStore": "1",
"connectionType": "Cloud",
"description": "Test",
"options": {
  "User": "mySForceUserId",
  "Password": "mySForcePassword",
  "SecurityToken": "mySecurityToken",
  "EnableBulkLoad": "true",
  "StmtCallLimit": "60",
  "MaxPooledStatements": "60"
},
"permissions": [
  1,
  2,
  3,
  4,
  5
]
}

Sample Server Response

Status code: 200
Successful response

{
  "id": "5039",
  "name": "SF2",
  "dataStore": "1",
  "connectionType": "Cloud",
  "description": "Test",
  "options": {
    "User": "mySForceUserId",
    "Password": "mySForcePassword",
    "SecurityToken": "mySecurityToken",
    "EnableBulkLoad": "true",
    "StmtCallLimit": "60",
    "MaxPooledStatements": "60"
  },
  "permissions": [
    1,
    2,
    3,
    4,
    5
  ]
}

Authentication

Basic Authentication using Login ID and Password.

Authorization

The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

See also

Create or Refresh a Data Source Model on page 1328
Delete a data source

Purpose
Deletesthespecifieddatasource.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}

Method
DELETE

URL Parameters
<myserver>isthehostnameorIPaddressofthemachinehostingtheHybridDataPipelineserverforastandaloneinstallation,orthemachinehostingtheloadbalancerforaloadbalancerinstallation. Forastandalone installation,<port>istheportnumberspecifiedastheServerAccessPortduringinstallation. Foraload balancerinstallation,<port>mustbeeither80forhttpor443forhttps. Wheneverport80or443areused,itis notnecessarytoincludetheportnumberintheURL.
The{datasourceId}parametermustalsobespecifiedintheURL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 204
Successful response

{
  "success":true
}

Sample Server Failure Response

{
  "error": {
    "code": "222207011",
    "message": {
      "lang": "en-US",
      "value": "Invalid DataSource ID: 1."
    }
  }
}
Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and DeleteDataSource (4) permissions.

Get data source permissions

Purpose
Retrieves the effective permissions on a data source. When permissions have not been explicitly set on the
data source, the effective permissions are the permissions of the user's role and any explicit permissions set
for the user. When permissions have been explicitly set on the data source, the effective permissions are the
same as the permissions that have been explicitly set regardless of role and user permissions.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter
to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/permissions

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a
standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone
installation, <port> is the port number specified as the Server Access Port during installation. For a load
balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it
is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
   "permissions": [integer, integer, ...]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of effective permissions. When permissions have not been explicitly set on the data source, the effective permissions are the permissions of the user's role and any explicit permissions set for the user. When permissions have been explicitly set on the data source, the effective permissions are the same as the permissions that have been explicitly set regardless of role and user permissions.</td>
<td>A comma separated list of permission ID See <a href="#">Data source permissions</a> on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>

Sample Server Response

```
Status code: 200
Successful response

{
   "permissions": [
      2,
      3,
      4,
      5
   ]
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

**Data source permissions**

Permissions can be specified on a data source in either of the following ways.

- When creating or updating a data source on behalf of a user, an administrator can set permissions on the data source.
- When sharing a data source, the data source owner must set permissions on the data source.

Any valid permissions specified on the data source will override the permissions of users that use the data source. The exact set of permissions must be specified on the data source as no permissions are inherited from the user or user's role. Permissions set on a group data source override permissions set on any of its member data sources.
The following permissions can be set on a data source.

**Note:** The SQLEditor permission cannot be set on a shared data source. It can only be set on a data source by an administrator on behalf of a user.

<table>
<thead>
<tr>
<th>Name</th>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ViewDataSource</td>
<td>2</td>
<td>The details of the data source may be viewed.</td>
</tr>
<tr>
<td>ModifyDataSource</td>
<td>3</td>
<td>The data source may be modified.</td>
</tr>
<tr>
<td>DeleteDataSource</td>
<td>4</td>
<td>The data source may be deleted.</td>
</tr>
<tr>
<td>UseDataSourceWithJDBC</td>
<td>5</td>
<td>The data source may be queried with the JDBC driver.</td>
</tr>
<tr>
<td>UseDataSourceWithODBC</td>
<td>6</td>
<td>The data source may be queried with the ODBC driver.</td>
</tr>
<tr>
<td>UseDataSourceWithOData</td>
<td>7</td>
<td>The data source may be queried with an OData application.</td>
</tr>
<tr>
<td>SQLEditor</td>
<td>10</td>
<td>The data source may be queried with the SQL Editor in the Web UI.</td>
</tr>
</tbody>
</table>

**Update permissions on a data source**

**Purpose**
Updates the permissions on a data source. This operation can only be executed by an administrator on behalf of a user by including the user query parameter in the request and specifying the user name.

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/permissions?user=<userName>

where `<userName>` is the name of the user for whom permissions on the data source are being updated. The user must be the owner of the data source.

**Method**

PUT

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The `{datasourceId}` parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{datasourceId}</code></td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Request Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
    "permissions": [integer, integer, ...]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of permissions associated explicitly with the data source. Permissions can only be set on a data source by an administrator when creating or updating the data source on behalf of a user. Any permissions specified for this data source will override the permissions for the user or the user's role that own this data source. You must specify the exact set of permissions that you want to set for this data source as no permissions are inherited from the user or user's role if permissions are specified on a data source. Permissions set on a group data source override permissions set on any of its member data sources.</td>
<td>Required</td>
<td>A comma separated list of permission ID See Data source permissions on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```json
{
    "permissions": [
        2,
        3,
        4,
        5
    ]
}
```

Sample Server Response

Status code: 200
Successful response

```json
{
    "permissions": [
        2,
        3,
        4,
        5
    ]
}
```
Authentication
Basic Authentication using Login ID and Password.

Authorization
Permissions: The administrator must have the Administrator (12) permission; or the administrator must have administrative access on the tenant to which the user belongs, the MgmtAPI (11) permission, the OnBehalfOf (21) permission, and the ModifyDataSource (3) permission.

See also
Create or Refresh a Data Source Model on page 1328

Test a connection to a data source

Purpose
Test whether a connection can be made to a specified data source.

Note: This API cannot be used on a group data source. If the test API is used on a group data source, an error is returned. To check connectivity on a group data source, test each member of the group data source.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/test

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition
If the user ID and password of the back end data store (for example, Oracle Database or Salesforce) are stored in the data source, then an empty JSON payload is required.

{ }
If the user ID and password of the back end data store are not stored in the data source, then they must be specified in the request payload. The payload has the following format.

```json
{
    "user": "data_store_user_id",
    "password": "data_store_user_password"
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;user&quot;</td>
<td>The user ID needed to connect to the back end data store (for example, Oracle Database or Salesforce)</td>
<td>A valid user ID for the back end data store</td>
</tr>
<tr>
<td>&quot;password&quot;</td>
<td>The user password needed to connect to the back end data store (for example, Oracle Database or Salesforce)</td>
<td>A valid user password for the back end data store</td>
</tr>
</tbody>
</table>

Sample Request Payload

```json
{
    "user": "MyDbId",
    "password": "MyDbSecret"
}
```

Sample Server Response

```
Status code: 200
Successful response
```

```json
{
    "success":true
}
```

Sample Server Failure Response

```json
{
    "error":{
        "code":222207028,
        "message":{
            "lang":"en-US",
            "value":"Missing 'userId' in payload."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password.
Authorization
The user must have the MgmtAPI (11) permission, the ViewDataSource (2) permission, and at least one query permission such as UseDataSourceWithJDBC (5), UseDataSourceWithODBC (6) or UseDataSourceWithOData (7).

Refresh a data source map

Purpose
Most non-relational data sources supported by Hybrid Data Pipeline maintain a map that defines how the non-relational object model is mapped to a set of relational tables with rows and columns. Issuing a POST request to the map resource allows this map to be refreshed or recreated. The map should be refreshed when a change has been made to the back end non-relational data model. The map should be recreated if the options used by the data source to generate the map are changed.

Important: This API only refreshes the relational map of non-relational data sources. See Create or refresh a data source OData model on page 1328 for details on refreshing an OData data model for a data source.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/map

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.

```
{
  "map": "setting"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;map&quot;</td>
<td>Specifies whether the relational map of the non-relational data store should be refreshed or recreated.</td>
<td>Required</td>
<td>&quot;refresh&quot;</td>
</tr>
</tbody>
</table>

If set to "refresh", the relational map of the non-relational data store is refreshed. The map should be refreshed when a change has been made to the back end data model.

If set to "recreate", the relational map of the non-relational data store is recreated. The map should be recreated if the options used by the data source to generate the map are changed. This call will also pick up any changes made to the back end data model.

Sample Request Payload

```
{
  "map": "refresh"
}
```

Sample Server Response

Status code: 200
Successful response

```
{
  "success":true
}
```

Sample Server Failure Response

```
{
  "error":{
    "code":222207029,
    "message":{
      "lang":"en-US",
      "value":"Expected values for model: refresh / none. Your value was False. Please try again with proper value."
    }
  }
}
```
**Authentication**
Basic Authentication using Login ID and Password.

**Authorization**
The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

**See also**
Create or refresh a data source OData model on page 1328

**Create or refresh a data source OData model**

**Purpose**
Data sources that are enabled to be accessed through OData maintain an OData data model. The OData model must be created when a new data source is created, or when the schema map for a data source is changed. Additionally, an OData model should be refreshed so that changes made to a data source schema are visible in the OData data model.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/model

**Method**
POST

**URL Parameters**
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Request Payload Definition**
The request takes the following format. The properties of the request are described in the table that follows.

**Note:** All properties are optional. Therefore, an empty payload ({} may be passed with the request.
"user": "data_store_user_id",
"password": "data_store_user_password",
"restart": boolean
}

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;user&quot;</td>
<td>The user ID needed to connect to the back end data store (for example, Oracle Database or Salesforce)</td>
<td>Optional</td>
<td>If the data source does not contain the user needed to connect to the back end data store, the user ID must be supplied in the payload.</td>
</tr>
<tr>
<td>&quot;password&quot;</td>
<td>The user password needed to connect to the back end data store (for example, Oracle Database or Salesforce)</td>
<td>Optional</td>
<td>If the data source does not contain the user needed to connect to the back end data store, the user password must be supplied in the payload.</td>
</tr>
<tr>
<td>&quot;restart&quot;</td>
<td>Specifies the behavior of the data access service when a create or refresh request is submitted while an OData model is being built for the specified data source. If &quot;restart&quot; is not set to true and a model is currently being generated, a 409 status error is returned, indicating that the OData Model for this data source is currently being built.</td>
<td>Optional</td>
<td>true</td>
</tr>
</tbody>
</table>

If set to true, any OData model that is currently being built is discarded and the Connectivity Service builds a new OData Model for the data source. If set to false and a model is currently being generated, a 409 status error is returned, indicating that the OData Model for this data source is currently being built.

Sample Request Payload

```json
{
    "user": "MyDbId",
    "password": "MyDbSecret",
    "restart": true
}
```

Sample Server Response

```json
{
    "success": true
}
```

Sample Server Failure Response

```json
{
    "error": {
        "code": 222207054,
        "message": {
            "lang": "en-US",
            "value": "Cannot start the OData Model Creation because it is currently running. Please see the documentation if you wish to restart the creation."
        }
    }
}
```
Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

See also
Refresh a data source map on page 1326

Check status of the OData model refresh

Purpose
Checks the current status of the refresh of the OData model. This call also returns information regarding tables and columns that were dropped while generating the OData Model for a given schema map of a Data Source. Since the OData model creation is asynchronous, all the warnings get stored in a table named ModelWarnings and these details are reported back when the user queries for a model status.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:@port/api/mgmt/datasources/{datasourceId}/model

Method
GET

URL Parameters
- <myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
- The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Response Definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
   "statusCode": status_number,
   "status": "status_message",
   ...
}
```

Depending on status, the following properties may be included in the response.

- **Model complete status**
  - "createdAt": "YYYY-MM-DD HH:mm:ss"
  - "tableWarnings": table_information
  - "columnWarnings": column_information

- **Working on model status**
  - "percentDone": "percent_done"

- **Problem status**
  - "reason": "message_on_refresh_error"

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;statusCode&quot;</td>
<td>Provides a code for the status of the refresh</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If -1, the model must be created before it can be refreshed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If 0, the refresh of the model is complete. The updated model is ready to use. The &quot;createdAt&quot; field shows the time at which the model was created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If 1, the model is currently being refreshed. The &quot;percentDone&quot; field shows the progress of the model refresh.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If 2, a problem was encountered. The &quot;reason&quot; field shows details about the problem.</td>
</tr>
<tr>
<td>&quot;status&quot;</td>
<td>A message that reports the status of the refresh</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Depending on the status of the refresh one the following messages are provided. The messages correspond to one of the four status codes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Model not created.(-1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Model is complete(0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Working on model.(1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>There was a problem creating the model.(2)</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;createdAt&quot;</td>
<td>The time at which the OData model was created</td>
<td>A timestamp in the UTC format YYYY-MM-DD HH:mm:ss provided if the OData model creation is complete</td>
</tr>
<tr>
<td>&quot;tableWarnings&quot;</td>
<td>Information on tables that were dropped from the data source schema while the OData model was generated</td>
<td>An array of table names with details on why the table was not included in the data source schema</td>
</tr>
<tr>
<td>&quot;columnWarnings&quot;</td>
<td>Information on columns that were dropped from the data source schema while the OData model was generated</td>
<td>An array of column names with their table names and details on why the column was not included in the data source schema</td>
</tr>
<tr>
<td>&quot;percentDone&quot;</td>
<td>A message that reports what percentage of the OData model creation has been completed</td>
<td>A string with percent done provided if the OData model creation is currently taking place</td>
</tr>
<tr>
<td>&quot;reason&quot;</td>
<td>A message that provides details about an error encountered during OData model creation</td>
<td>A string with error message details provided if the OData model creation has encountered an error</td>
</tr>
</tbody>
</table>

Sample Server Response

**Example 1:** Model creation is proceeding correctly.

```json
{
   "statusCode": 1,
   "status": "Working on model.",
   "percentDone": "80 percent done"
}
```

**Example 2:** Model creation is complete

```json
{
   "statusCode": 0,
   "status": "Model is complete.",
   "createdAt": "2017-07-17 09:25:12.812",
   "tableWarnings": [
      {
         "table": "NOPRIMARYLONG",
         "reason": "No primary key has been specified for this table."
      }
   ],
   "columnWarnings": [
      {
         "table": "BOOKS1",
         "column": "SENTENCE",
         "reason": "The column size is too long. Actual size is 2,147,483,647 and supported size is 32,768."
      },
      {
         "table": "NOPRIMARYLONG",
         "column": "LONGCOL",
         "reason": "The column size is too long. Actual size is 2,147,483,647 and supported size is 32,768."
      }
   ]
}
```
Sample Server Failure Response

When the refresh operation is not proceeding correctly, a response similar to the following is returned:

```
{
  "statusCode": 2,
  "status": "There was a problem creating the model.",
  "reason": "No primary key"
}
```

Authentication

Basic Authentication using Login ID and Password.

Authorization

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

See also

Create or Refresh a Data Source Model on page 1328

Get members of a data source group

Purpose

Returns the member data sources for the group data source.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL

https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {groupDatasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{groupDatasourceId}</td>
<td>The ID of the group data source.</td>
<td>The ID is auto-generated when the group data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Response Definition

The request takes the following format. The properties of the request are described in the table that follows.
Note: The members object is an array that contains one or more member data sources. Each data source
must have an ID and an entity prefix.

```json
{
    "members": [
        {
            "id": idnum1,
            "entityPrefix": "prefix1"
        },
        {
            "id": idnum2,
            "entityPrefix": "prefix2"
        }
    ]
}
```

If the group data source has no members, the response is an empty list.

```json
{
    "members": []
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the member data source that belongs to the group data source. The ID is auto-generated when the data source is created and cannot be changed. A member data source cannot itself be a group data source.</td>
<td></td>
</tr>
<tr>
<td>&quot;entityPrefix&quot;</td>
<td>A user-defined prefix associated with a specific data source to resolve naming conflicts. This prefix is added to all tables that come from the specified data source. For example, suppose a member data source is specified with the prefix acct and the data source has a table named customers. This table is identified by the name acct_customers in the group data source.</td>
<td>The prefix must be 1 to 64 characters in length and must be unique.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 200
Successful response

```json
{
    "members": [
        {
            "id": 3,
            "entityPrefix": "fin"
        },
        {
```
Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

Add member data sources to a group data source group

Purpose
Add one or more member data sources to a group data source.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {groupDatasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{groupDatasourceId}</td>
<td>The ID of the group data source.</td>
<td>The ID is auto-generated when the group data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.
Note: The members object is an array that contains one or more member data sources. Each data source must have an ID and an entity prefix.

```json
{
   "members": [
      {
         "id": "idnum1",
         "entityPrefix": "prefix1"
      },
      ...  
   ]
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the member data source that belongs to the group data source</td>
<td>Required</td>
<td>The ID is auto-generated when the data source is created and cannot be changed. A member data source cannot itself be a group data source.</td>
</tr>
<tr>
<td>&quot;entityPrefix&quot;</td>
<td>A user-defined prefix associated with a specific data source to resolve naming conflicts. This prefix is added to all tables that come from the specified data source. For example, suppose a member data source is specified with the prefix acct and the data source has a table named customers. This table is identified by the name acct_customers in the group data source.</td>
<td>Required</td>
<td>The prefix must be 1 to 64 characters in length and should be unique.</td>
</tr>
</tbody>
</table>

Sample Payload Request

```json
{
   "members": [
      {
         "id": 11,
         "entityPrefix": "sal"
      }
   ]
}
```

Sample Server Response

```json
{
   "success": true
}
```

Authentication

Basic Authentication using Login ID and Password.
Authorization
The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

Update members of a group data source

Purpose
Updates the member data sources that comprise a group data source. The member data sources provided in the payload replace the data sources currently specified as members of the group data source. The member data sources specified must be owned by the same user that owns the data source group.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members

Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {groupDatasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{groupDatasourceId}</td>
<td>The ID of the group data source.</td>
<td>The ID is auto-generated when the group data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Request Payload Definition
The request takes the following format. The properties of the request are described in the table that follows.

Note: The members object is an array that contains one or more member data sources. Each data source must have an ID and an entity prefix.

```json
{
    "members": [
        {
            "id": idnum1,
            "entityPrefix": "prefix1"
        },
        {
            "id": idnum2,
            "entityPrefix": "prefix2"
        }
    ]
}```
To remove all the members from a group, an empty list may be passed.

```json
{
    "members": []
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the member data source that belongs to the group data source</td>
<td>Required</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A member data source cannot itself be a group data source.</td>
</tr>
<tr>
<td>&quot;entityPrefix&quot;</td>
<td>A user-defined prefix associated with a specific data source to resolve naming conflicts.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This prefix is added to all tables that come from the specified data source.</td>
<td>Required</td>
<td>The prefix must be 1 to 64 characters in length and must be unique.</td>
</tr>
<tr>
<td></td>
<td>For example, suppose a member data source is specified with the prefix acct and the data source has a table named customers. This table is identified by the name acct_customers in the group data source.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```json
{
    "members": [
        {
            "id": 3,
            "entityPrefix": "fin"
        },
        {
            "id": 6,
            "entityPrefix": "mkt"
        },
        {
            "id": 11,
            "entityPrefix": "sal"
        }
    ]
}
```

**Sample Server Response**

```json
{
    "success": true
}
```
Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

Delete a member data source from a group data source

Purpose
Removes a member data source from a group data source group.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{groupDatasourceId}/members/{memberDatasourceId}

Method
DELETE

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {groupDatasourceId} and {memberDatasourceId} parameters must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{groupDatasourceId}</td>
<td>The ID of the group data source.</td>
<td>The ID is auto-generated when the group data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{memberDatasourceId}</td>
<td>The ID of the member data source.</td>
<td>The ID is auto-generated when the member data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 204
Successful response

```json
{
    "success":true
}
```
Sample Server Failure Response

```json
{
    "error": {
        "code": "222207004",
        "message": {
            "lang": "en-US",
            "value": "There is no DataSource with that id: 5038."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

**Get shared data source users**

**Purpose**

Retrieves users with whom the data source is being shared.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**

https://<myserver>:<port>/api/mgmt(datasources/<datasourceId>/sharedUsers

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a user or users.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Response Definition

The response takes the following format.

```
{
    "sharedUsers": [
    {
        "userId": "user_id",
        "permissions": [
            permission,
            permission,
            ...
        ],
    },
    ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userId&quot;</td>
<td>The ID of the user account with which the data source is being shared.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of data source permissions granted to the shared user account. The shared user will only be able to execute operations against the data source that correspond to the permissions granted. The data source owner must specify the exact set of permissions for shared users as no permissions are inherited from the user or user's role.</td>
<td>A comma separated list of permission IDs. See Data source permissions on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

```
Status code: 200
Successful response

{
    "sharedUsers": [
    {
        "userId": 88,
        "permissions": [2, 3, 5]
    },
    {
        "userId": 89,
        "permissions": [2, 3, 5]
    }
    ]
}
```
Sample Server Failure Response

```json
{
    "error": {
        "code": 222207093,
        "message": {
            "lang": "en-US",
            "value": "DataSource 5441 is not shared with any users."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

**Share data source with a user or users**

**Purpose**

Shares a data source with a user or users.

---

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

---

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers

**Method**

POST

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a user or users.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
## Request Payload Definition

The request takes the following format.

```json
{
    "sharedUsers": [
        {
            "userId": user_id,
            "permissions": [
                permission,
                permission,
                ...
            ],
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userId&quot;</td>
<td>The ID of the user account with which the data source is being shared.</td>
<td>Required</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of data source permissions granted to the shared user account. The shared user will only be able to execute operations against the data source that correspond to the permissions granted. The data source owner must specify the exact set of permissions for shared users as no permissions are inherited from the user or user's role.</td>
<td>Required</td>
<td>A comma separated list of permission IDs. See Data source permissions on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>

## Request Payload

```json
{
    "sharedUsers": [
        {
            "userId": 88,
            "permissions": [
                2,
                3,
                5
            ]
        },
        {
            "userId": 89,
            "permissions": [
                2,
                3,
                5
            ]
        }
    ]
}
```
Success Response

Status code: 201
Successful response

```
{
    "sharedUsers": [
        {
            "userId": 88,
            "permissions": [
                2,
                3,
                5
            ]
        },
        {
            "userId": 89,
            "permissions": [
                2,
                3,
                5
            ]
        }
    ]
}
```

Authentication

Basic Authentication using Login ID and Password.

Authorization

The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

Delete shared users from the data source

Purpose

Stops sharing the data source with users. This operation will delete shared users from the data source.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a username. See also *Managing resources on behalf of users* on page 1281.

URL

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers

Method

DELETE

URL Parameters

*<myserver>* is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, *<port>* is the port number specified as the Server Access Port during installation. For a load balancer installation, *<port>* must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The `{datasourceId}` parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{datasourceId}</code></td>
<td>The ID of the data source being shared with a user or users.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Sample Server Response**

```
Status code: 204
Successful response

{
  "success":true
}
```

**Sample Server Failure Response**

```
{
  "error": {
    "code": "222207011",
    "message": {
      "lang": "en-US",
      "value": "Invalid DataSource ID: 1."
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and DeleteDataSource (4) permissions.

**Get the data source permissions for a shared user**

**Purpose**

Retrieves the data source permissions for a user with whom the data source is being shared.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers/{userId}

**Method**

GET
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} and {userId} parameters must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a user or users.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{userId}</td>
<td>The ID of the user with whom the data source is being shared.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Response Definition

The response takes the following format.

```json
{
    "permissions": [
        [
            "permission",
            "permission",
            ...
        ]
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of data source permissions granted to the shared user account.</td>
<td>A comma separated list of permission IDs.</td>
</tr>
<tr>
<td></td>
<td>The shared user will only be able to execute operations against the data</td>
<td>See Data source permissions on page 1321 for supported permissions.</td>
</tr>
<tr>
<td></td>
<td>source that correspond to the permissions granted.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The data source owner must specify the exact set of permissions for shared</td>
<td></td>
</tr>
<tr>
<td></td>
<td>users as no permissions are inherited from the user or user's role.</td>
<td></td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```json
{
    "permissions": [
        2,
        ...
    ]
}
```
Sample Server Failure Response

```json
{
    "error": {
        "code": 222207093,
        "message": {
            "lang": "en-US",
            "value": "DataSource 5441 is not shared with any users."
        }
    }
}
```

**Authentication**
Basic Authentication using Login ID and Password

**Authorization**
The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

**Update data source permissions for shared user**

**Purpose**
Updates the data source permissions for a user with whom the data source is being shared.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**
`https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers/{userId}`

**Method**
PUT

**URL Parameters**
- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
- The `{datasourceId}` and `{userId}` parameters must also be specified in the URL.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a user or users.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{userId}</td>
<td>The ID of the user with whom the data source is being shared.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Request Payload Definition**

The request takes the following format.

```json
{
    "permissions": [
        [permission, permission, ...]
    ]
}
```

**Property** | **Description** | **Valid Values**
--- | --- | ---
"permissions" | A list of data source permissions granted to the shared user account. The shared user will only be able to execute operations against the data source that correspond to the permissions granted. The data source owner must specify the exact set of permissions for shared users as no permissions are inherited from the user or user's role. | A comma separated list of permission IDs. See Data source permissions on page 1321 for supported permissions.

**Sample Request Payload**

```json
{
    "permissions": [
        [2, 3, 4, 5]
    ]
}
```

**Sample Server Response**

Status code: 200
Successful response
{ "permissions": [ [ 2, 3, 4, 5 ] ] }

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

**Delete shared user from a data source**

**Purpose**

 Stops sharing the data source with a user. This operation will delete the shared user from the data source.

---

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

---

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedUsers/{userId}

**Method**

DELETE

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} and {userId} parameters must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a user or users.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{userId}</td>
<td>The ID of the user with whom the data source is being shared.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Sample Server Response

Status code: 204
Successful response

{
  "success":true
}

Sample Server Failure Response

{
  "error": {
    "code": "222207011",
    "message": {
      "lang": "en-US",
      "value": "Invalid DataSource ID: 1."
    }
  }
}

Authentication

Basic Authentication using Login ID and Password.

Authorization

The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

Get shared data source tenants

Purpose

Retrieves tenants with which the data source is being shared.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The `{datasourceId}` parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{datasourceId}</code></td>
<td>The ID of the data source being shared with a tenant or tenants.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Response Definition**

The response takes the following format.

```json
{
    "sharedTenants": [
        {
            "tenantId": tenant_id,
            "permissions": [
                permission,
                permission,
                ...
            ],
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant with which the data source is being shared.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of data source permissions granted to all user accounts which belong to the tenant. The users in the tenant will only be able to execute operations against the data source that correspond to the permissions granted. The data source owner must specify the exact set of permissions as no permissions are inherited from the users or users’ roles.</td>
<td>A comma separated list of permission IDs. See <a href="#">Data source permissions</a> on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

Status code: 200  
Successful response

```json
{
    "sharedTenants": [
        {
            "tenantId": 12,
            "permissions": [
                2,
                3,
                5
            ]
        }
    ]
}
```
Sample Server Failure Response

```json
{
   "error": {
      "code": 222206951,
      "message": {
         "lang": "en-US",
         "value": "DataSource 431 is not shared with any tenants."
      }
   }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

The data source owner must have either the Administrator (12) permission; or the MgmtAPI (11) permission, the ViewDataSource (2) permission, and administrative access on the tenant with which the data source is being shared.

**Share data source with a tenant or tenants**

**Purpose**

Shares data source with a tenant or tenants.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{dataSourceId}/sharedTenants

**Method**

POST

**URL Parameters**

- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The `{datasourceId}` parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{datasourceId}</code></td>
<td>The ID of the data source being shared with a tenant or tenants.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Request Payload Definition**

The request takes the following format.

```json
{
    "sharedTenants": [
        {
            "tenantId": tenant_id,
            "permissions": [
                permission,
                permission,
                ...
            ],
        },
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant with which the data source is being shared.</td>
<td>Required</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of data source permissions granted to all user accounts which belong to the tenant. The users in the tenant will only be able to execute operations against the data source that correspond to the permissions granted. The data source owner must specify the exact set of permissions as no permissions are inherited from the users or users' roles.</td>
<td>Required</td>
<td>A comma separated list of permission IDs. See Data source permissions on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>

**Request Payload**

```json
{
    "sharedTenants": [
        {
            "tenantId": 12,
            "permissions": [
                2,
                3,
                5
            ],
        },
        {
            "tenantId": 25,
```
"permissions": [ 2, 3, 5 ]
]
]

Success Response

Status code: 201
Successful response

{
  "sharedTenants": [
    {
      "tenantId": 12,
      "permissions": [ 2, 3, 5 ]
    },
    {
      "tenantId": 25,
      "permissions": [ 2, 3, 5 ]
    }
  ]
}

Authentication

Basic Authentication using Login ID and Password.

Authorization

The data source owner must have either the Administrator (12) permission; or the MgmtAPI (11) permission, the ModifyDataSource (3) permission, and administrative access on the tenant with which the data source is being shared.

Delete shared tenants from a data source

Purpose

Stop sharing the data source with tenants. This operation will delete shared tenants from the data source.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a username. See also Managing resources on behalf of users on page 1281.

URL

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants
Method
DELETE

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a tenant or tenants.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 204
Successful response

```json
{
  "success":true
}
```

Sample Server Failure Response

```json
{
  "error": {
    "code": "222207011",
    "message": {
      "lang": "en-US",
      "value": "Invalid DataSource ID: 1."
    }
  }
}
```

Authentication
Basic Authentication using Login ID and Password.

Authorization
The data source owner must have either the Administrator (12) permission; or the MgmtAPI (11) permission, the ModifyDataSource (3) permission, and administrative access on the tenant with which the data source is being shared.

Get the data source permissions for a shared tenant

Purpose
Retrieves the data source permissions for a tenant with which the data source is being shared.
**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants/{tenantId}

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {datasourceId} and {tenantId} parameters must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a tenant or tenants.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{tenantId}</td>
<td>The ID of the tenant with which the data source is being shared.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Response Definition**

The response takes the following format.

```json
{
    "permissions": [
        permission,
        permission,
        ...
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of data source permissions granted to all user accounts which belong to the tenant. The users in the tenant will only be able to execute operations against the data source that correspond to the permissions granted. The data source owner must specify the exact set of permissions as no permissions are inherited from the users or users' roles.</td>
<td>A comma separated list of permission IDs. See Data source permissions on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

Status code: 200
Successful response

```json
{
  "permissions": [
    [2, 3, 5]
  ]
}
```

Sample Server Failure Response

```json
{
  "error": {
    "code": 222206951,
    "message": {
      "lang": "en-US",
      "value": "DataSource 431 is not shared with any tenants."
    }
  }
}
```

Authentication
Basic Authentication using Login ID and Password

Authorization
The data source owner must have either the Administrator (12) permission; or the MgmtAPI (11) permission, the ViewDataSource (2) permission, and administrative access on the tenant with which the data source is being shared.

Update data source permissions for shared tenant

Purpose
Update the data source permissions for a tenant with which the data source is being shared.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants/{tenantId}

Method
PUT
**URL Parameters**

*<myserver>* is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, *<port>* is the port number specified as the Server Access Port during installation. For a load balancer installation, *<port>* must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {*datasourceId}* and {*tenantId}* parameters must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{<em>datasourceId}</em></td>
<td>The ID of the data source being shared with a tenant or tenants.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{<em>tenantId}</em></td>
<td>The ID of the tenant with which the data source is being shared.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Request Payload Definition**

The request takes the following format.

```
{
  "permissions": [
    [permission,
    permission,
    ...
  ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of data source permissions granted to all user accounts which belong to the tenant. The users in the tenant will only be able to execute operations against the data source that correspond to the permissions granted. The data source owner must specify the exact set of permissions as no permissions are inherited from the users or users’ roles.</td>
<td>A comma separated list of permission IDs. See Data source permissions on page 1321 for supported permissions.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```
{
  "permissions": [
    [2,
    3,
    4,
    5
  ]
}
```

Sample Server Response

Status code: 200
Successful response

```json
{
  "permissions": [
    [2, 3, 4, 5]
  ]
}
```

Authentication
Basic Authentication using Login ID and Password.

Authorization
The data source owner must have either the Administrator (12) permission; or the MgmtAPI (11) permission, the ModifyDataSource (3) permission, and administrative access on the tenant with which the data source is being shared.

Delete shared tenant from a data source

Purpose
 Stops sharing the data source with a tenant. This operation will delete the shared tenant from the data source.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL

https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/sharedTenants/{tenantId}

Method
DELETE

URL Parameters

- `<myserver>` is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, `<port>` is the port number specified as the Server Access Port during installation. For a load balancer installation, `<port>` must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

- The `{datasourceId}` and `{tenantId}` parameters must also be specified in the URL.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{datasourceId}</td>
<td>The ID of the data source being shared with a tenant or tenants.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>{tenantId}</td>
<td>The ID of the tenant with which the data source is being shared.</td>
<td>The ID is auto-generated when the tenant is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 204
Successful response

```
{
    "success": true
}
```

Sample Server Failure Response

```
{
    "error": {
        "code": "222207011",
        "message": {
            "lang": "en-US",
            "value": "Invalid DataSource ID: 1."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password.

Authorization

The data source owner must have either the Administrator (12) permission; or the MgmtAPI (11) permission, the ModifyDataSource (3) permission, and administrative access on the tenant with which the data source is being shared.

Driver Files API

The Driver Files API is an extension of the Data Sources API. The Driver Files API can be used for the following purposes.

- Export schema map files for non-relational data sources
- Manage input and output REST files for REST data sources
Export schema map files for non-relational data sources

When initially connecting to a data source for a non-relational data store such as Salesforce, the connectivity service creates a pair of schema map files. These files include the native file and the config file. The native file is an XML file that describes the object model of a non-relational data store. The config file is an XML file that exposes the object model as a set of relational tables with rows and columns. Together these files support SQL queries to non-relational data stores. These files can be useful in developing valid SQL statements and in troubleshooting issues that may arise when querying non-relational data stores. The following operations allow you to export these files.

**Note:** The Driver Files API cannot be used to retrieve files when the On-Premises Connector is used to connect to a web service (or non-relational data store) such as Salesforce.

<table>
<thead>
<tr>
<th>Task</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export driver files for data source</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{id}/export/driverfiles</td>
</tr>
<tr>
<td>Export config files for data source</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{id}/export/driverfiles/config</td>
</tr>
<tr>
<td>Export native files for data source</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{id}/export/driverfiles/native</td>
</tr>
</tbody>
</table>

Manage input and output REST files for REST data sources

A Hybrid Data Pipeline REST data source can be created by way of the Autonomous REST Connector data store. A REST data source must include the specification REST endpoints either via the Web UI or by uploading an input REST file. The input REST file is a JSON file which specifies one or more REST endpoints in the form of a JSON object (see Creating an input REST file on page 641 for syntax requirements). The input REST file can be retrieved and managed using API requests listed in the table below.

When initially connecting to a REST endpoint, Hybrid Data Pipeline uses the input REST file to build a relational model of the REST data. This model is used to translate and execute SQL queries against the REST service, and it is available in the form of the output REST file. Therefore, a review of the output REST file may be useful for developing an input REST file and creating better SQL queries. The output REST file cannot be edited directly. Note that an initial connection to the REST service must be made before the output REST file is available.

**Note:** For an overview on REST connectivity, see Creating and using REST data sources on page 637.

<table>
<thead>
<tr>
<th>Task</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve input REST file</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{id}/export/driverfiles/inputrest</td>
</tr>
<tr>
<td>Upload input REST file</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{id}/export/driverfiles/inputrest</td>
</tr>
</tbody>
</table>
### Export driver files for data source

**Purpose**
Exports the driver files for a specified non-relational data source. The response file is streamed to the user, who can then download the artifacts as a zip file.

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {id} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

Status code: 200
Successful response

**Sample Server Failure Response**

```json
{
    "error": {
        "code": 222208729,
        "message": "No Schema map files found."
    }
}
```
Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ViewDataSource (2) permission on the applicable data source.

Export config files for data source

Purpose
Exports the config file for a specified non-relational data source. The file will be returned as an XML response.

URL
https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles/config

Method
GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {id} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 200
Successful response

```xml
<?xml version='1.0' encoding='UTF-8'?><Database xmlns="http://test-datadirect.com/cloud/config" version="2">
  <User name="**********" defaultSchema="*****">
    <UseSchema name="*****"/>
    <UseSchema name="PUBLIC"/>
  </User>
  <Map name="*****" type="Eloqua">
    <ConfigOptions>UPPERCASEIDENTIFIERS=1;...;...</ConfigOptions>
    <SessionOptions>DATABASENAME=;USER=;...;...</SessionOptions>
  </Map>
  <MapDatabase uppercaseidentifiers="true" ... truncateMethod="asis">
    <Schema native="ELOQUA" rename="ELOQUA" default="true">
      <Table native="Account" rename="ACCOUNT">
        <Column rename="ID" path="id/*" key="1" dataType="LONG"/>
```
Sample Server Failure Response

```json
{
    "error": {
        "code": 222208728,
        "message": "No config cloud db driver file found."
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ViewDataSource (2) permission on the applicable data source.

**Export native file for data source**

**Purpose**

Exports the native file for the specified non-relational data source. The file will be shown as an XML response.

**URL**

https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles/native

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {id} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
Sample Server Success Response

Status code: 200
Successful response

<?xml version='1.0' encoding='UTF-8'?>
<Native xmlns="http://test-datadirect.com/cloud/native" version="11" nativeVersion="1"
xmlns:z="http://test-datadirect.com/cloud/native/sforce">
  <Options>
    <OptionSet name="Set1">
      <Option value="a__c"/>
      <Option value="BigTable__c"/>
      <Option value="BINTABLE__c"/>
      <Option value="BITABLE__c"/>
      <Option value="BTABLE__c"/>
      ...
    </OptionSet>
    ...
  </Options>
  <Packages>
    <Package name="SFORCE">
      <Object name="AcceptedEventRelation" label="Accepted Event Relation">
        <Fields>
          <Field name="Id" label="Event Relation ID"/>
          <Field name="RelationId" label="Relation ID"/>
          <Field name="EventId" label="Event ID"/>
          ...
        </Fields>
        <Parents>
          <Parent name="Relation0" parentKeyPath="Id/*"/>
          <KeyPart path="RelationId/*" parentKeyPath="Id/*"/>
          <Parent name="Relation1" parentPackage="SFORCE">
            <KeyPart path="RelationId/*" parentKeyPath="Id/*"/>
            ...
          </Parent>
          ...
        </Parents>
        <Children>
          <Child name="ChildAccounts" childPackage="SFORCE">
            <KeyPart path="Id/*" childKeyPath="ParentId/*"/>
          </Child>
          <Child name="AccountContactRoles" cascadeDelete="true">
            <KeyPart path="Id/*" childKeyPath="AccountId/*"/>
          </Child>
          ...
        </Children>
      </Object>
    </Package>
  </Packages>
</Native>

Sample Server Failure Response

{  "error":  {  "code": 222208728,  "message": "No native cloud db driver file found."  }}

Authentication

Basic Authentication using Login ID and Password.
Authorization
The user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ViewDataSource (2) permission on the applicable data source.

Get input REST file

Purpose
Retrieves the input REST file. The REST file is a JSON object provided in the response payload.

URL
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/export/driverfiles/inputrest

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {id} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

{
    "countries": {
        "#path": "http://example.com/country",
        "#post": {
            "start_date":"2018-08-31",
            "end_date":"2018-09-01",
            "departments":["engineering,marketing, sales"],
            "tags":["blue, green, red"]
        }
    }
}

Sample Server Failure Response

{
"error": {
  "code": 222208734,
  "message": {
    "lang": "en-US",
    "value": "inputrest driver file is not applicable for datasources with datastore id {1}. Applicable datastore id is 62."
  }
}

Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ViewDataSource (2) permission on the applicable data source.

Upload input REST file

Purpose
Uploads the input REST file. The REST file must be provided in the form of a JSON object in the request payload. For syntax requirements, see Creating an input REST file on page 641.

URL
https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles/inputrest

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {id} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```
{
  "countries": {
    "#path": "http://example.com/country",
    "#get": {
      "start_date": "2018-08-31",
      "end_date": "2018-09-01",
```
Chapter 10: Hybrid Data Pipeline API reference

"departments": "[engineering, marketing, sales]",
"tags": "[blue, green, red]"
}
}

Sample Server Success Response

Status code: 201
Successful response

{
  "countries": {
    "#path": "http://example.com/country",
    "#get": {
      "start_date": "2018-08-31",
      "end_date": "2018-09-01",
      "departments": "[engineering, marketing, sales]",
      "tags": "[blue, green, red]"
    }
  }
}

Sample Server Failure Response

{
  "error": {
    "code": 222208734,
    "message": {
      "lang": "en-US",
      "value": "inputrest driver file is not applicable for datasources with datastore id {1}. Applicable datastore id is 62."
    }
  }
}

Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ModifyDataSource (3) permission on the applicable data source.

Update input REST file

Purpose
Updates the input REST file. The updated REST file must be provided in the form of a JSON object provided in the request payload. For syntax requirements, see Creating an input REST file on page 641.

URL
https://<myserver>:<port>/api/mgmt/datasources/{id}/export/driverfiles/inputrest
Method
PUT

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {id} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```
{
  "countries": {
    "#path": "http://example.com/country",
    "#post": {
      "start_date": "2018-10-01",
      "end_date": "2018-10-31",
      "departments": "[engineering,marketing,sales]",
      "tags": "[blue,green,red]"
    }
  }
}
```

Sample Server Success Response

Status code: 200
Successful response

```
{
  "countries": {
    "#path": "http://example.com/country",
    "#post": {
      "start_date": "2018-10-01",
      "end_date": "2018-10-31",
      "departments": "[engineering,marketing,sales]",
      "tags": "[blue,green,red]"
    }
  }
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222208734,
    "message": {
      "lang": "en-US",
```
"value": "inputrest driver file is not applicable for datasources with datastore id {1}. Applicable datastore id is 62."

Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ModifyDataSource (3) permission on the applicable data source.

Get output REST file

Purpose
Retrieves the output REST file. The output REST file is a JSON object provided in the response payload.

URL
https://<myserver>:<port>/api/mgmt/datasources/{datasourceId}/export/driverfiles/outputrest

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The {id} parameter must also be specified in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the data source.</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```json
{
  "countries": {
    "#path": [
      "https://example.com/country"
    ],
    "type": "VarChar(64),#key",
    "metadata": {
```

Sample Server Failure Response

```
{
    "error": {
        "code": 222208734,
        "message": {
            "lang": "en-US",
            "value": "inputrest driver file is not applicable for datasources with datastore id {1}. Applicable datastore id is 62."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have either the Administrator (12) permission, or the MgmtAPI (11) permission and ViewDataSource (2) permission on the applicable data source.

**Management Permissions API**

The Management Permissions API is part of the Hybrid Data Pipeline Management API. The Management Permissions API allows a user to retrieve the effective permissions on a user account. The permissions for a user account are the sum of the permissions granted to the role(s) associated with the account and the permissions granted explicitly on the account. Any permissions specified on a data source will override the permissions for the user that owns the data source. (See also User provisioning on page 107.)

The following operation is supported with the Management Permissions API.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve the user's permissions</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/permissions</td>
</tr>
</tbody>
</table>
Get permissions on a user account

Purpose
Retrieves the user's permissions.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/permissions

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition
The response takes the following format.

```json
{
    "userId": user_account_id,
    "permissions": [permission_id, permission_id, ...]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;userId&quot;</td>
<td>The ID of the user account.</td>
<td>The ID is auto-generated when the user account is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;permissions&quot;</td>
<td>A list of effective permissions granted to the user account. Effective permissions for a user account are the sum of the permissions granted to the role(s) associated with the account and the permissions granted explicitly on the account.</td>
<td>A comma separated list of permission IDs. See Permissions and default roles on page 59 for a list of supported permissions.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```json
{
    "userId": 86,
    "permissions": [permission_id, permission_id, ...]
}
```
"permissions": [
  1,
  2,
  3,
  4,
  5,
  6,
  7,
  8,
  9,
  10,
  11,
  12
]
}

Sample Server Failure Response
{
  "error":{
    "code":222207031,
    "message":{
      "lang":"en-US",
      "value":"Invalid Progress ID userName."  
    }
  }
}

Authentication
Basic Authentication using Login ID and Password

Authorization
The user must have the MgmtAPI (11) permission.

OAuth API for configuring Hybrid Data Pipeline to authorize client applications

To support OAuth 2.0 authentication, you can register your application with Hybrid Data Pipeline.

The Client Application Registration API can be used to grant client applications access to Hybrid Data Pipeline data sources using OAuth 2.0 authentication. With the Client Application Registration API, you can register a client application with Hybrid Data Pipeline to generate a client ID and client secret. The client ID and client secret can then be used to generate tokens that enable applications to authenticate against Hybrid Data Pipeline with OAuth 2.0. You can also use the APIs to view a list of registered applications, reset client credentials, revoke access to a registered application, and otherwise manage client application access to Hybrid Data Pipeline data sources using OAuth 2.0.

The following table summarizes the operations that can be carried out with the set of APIs.
<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get list of OAuth registered applications</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications</td>
</tr>
<tr>
<td>Register OAuth application</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications</td>
</tr>
<tr>
<td>Get registered application by ID</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}</td>
</tr>
<tr>
<td>Update registered application by ID</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}</td>
</tr>
<tr>
<td>Delete registered application by ID</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}</td>
</tr>
<tr>
<td>Reset client secret of application</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/applications/{id}/reset</td>
</tr>
<tr>
<td>Get list of applications for which logged-in user has access</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth/client/allowedapplications</td>
</tr>
<tr>
<td>Revoke access granted for the given application ID</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oath/client/allowedapplications/{id}</td>
</tr>
<tr>
<td>Generate access token and refresh token</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth2/token</td>
</tr>
<tr>
<td>Authorize token</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauth2/authorize</td>
</tr>
</tbody>
</table>

**Get list of OAuth registered applications**

**Purpose**

Returns list of OAuth registered applications

**URL**

https://<myserver>:<port>/api/mgmt/oauth/client/applications

**Method**

GET
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition

The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
    "applications": [
        {
            "id": "app_id",
            "name": "app_name",
            "description": "app_description",
            "redirectUrls": [
                "redirect_url1", "redirect_url2", ...
            ]
        }
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is an integer. It is automatically generated with the successful registration of the application.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The user-specified name of the application.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>The user-specified description of the application.</td>
</tr>
<tr>
<td>&quot;redirectUrls&quot;</td>
<td>List of authorized URLs specified by the client. These are the URLs that the application should redirect to, on successful authorization. This may be one valid URL or a comma separated list of valid URLs.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

```
{
    "id": 19,
    "name": "Application1",
    "description": "Application1 for Create with all Fields",
    "redirectUrls": ["bedford.progresstest.com", "americas.progresstest.com"],
}
```

Sample Server Failure Response

```
{
    "error": {
        "code": 222206631,
        "message": {
            "lang": "en-US",
```

"value":"Problem getting OAuth Client Application at this time. Please try again at another time."
}
}

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

Any active Hybrid Data Pipeline user

**Register OAuth application**

**Purpose**

Registers OAuth application. The execution of this request results in the generation of a client ID and client secret required for OAuth authentication.

**URL**

https://<myserver>:<port>/api/mgmt/oauth/client/applications

**Method**

POST

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Request Payload Definition**

The request payload is a JSON object with the following format:

```
{
    "applications" :
    {
    {
        "id": "app_id",
        "name": "app_name",
        "description": "app_description",
        "redirectUrls": [
            "redirect_url1",
            "redirect_url2",
            ...
        ]
    }
    }
}
```

**Valid Values Description Property**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>User specified name of the application.</td>
<td>A string with a maximum length of 128 characters.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>User specified description of the application.</td>
<td>A string with a maximum length of 256 characters</td>
</tr>
<tr>
<td>&quot;redirectUrls&quot;</td>
<td>User defined list of authorized URLs specified by the client. These are the URLs that the application should redirect to, on successful authorization.</td>
<td>One or more valid URLs. You can enter multiple URLs, separated by commas.</td>
</tr>
</tbody>
</table>

**Sample Payload**

```
{
    "name": "Application1",
    "description": "Application1 for Create with all Fields",
    "redirectUrls": ["bedford.progresstest.com", "americas.progresstest.com"]
}
```

**Response Definition**

When the request is executed, a client ID and a client secret are generated. The parameters of the response are described in the table that follows.

```
{
    "id": app_id,
    "name": app_name,
    "description": "app_description"
    "redirectUrls": ["redirect_url1", "redirect_url2",...],
    "clientId": "string",
    "clientSecret": "string"
}
```

**Property**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application. This ID is used for tracking applications in Hybrid Data Pipeline.</td>
<td>An auto-generated application ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the application</td>
<td>A string with a maximum length of 128 characters.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>User specified description of the application.</td>
<td>A string with a maximum length of 256 characters</td>
</tr>
<tr>
<td>&quot;redirectUrls&quot;</td>
<td>List of authorized URLs</td>
<td>This may be one valid URL or a comma separated list of valid URLs.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The client ID is generated when the client application is registered. This ID is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The client secret is generated when the client application is registered. This secret is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
</tbody>
</table>

**Sample Server Success Response**

Status code: 201
Created

```
{
    "id":19,
    "name":"Application1",
    "description":"Application1 for Create with all Fields",
    "redirectUrls":["bedford.progressstest.com","americas.progressstest.com"],
    "clientId":"315368974.apps.hdptest.com",
    "clientSecret":"96dab351-cd80-4dfc-8756-8afe9896e92f"
}``
```

**Sample Server Failure Response**

```
{
    "error":{
        "code":222206628,
        "message":{
            "lang":"en-US",
            "value":"Problem creating OAuth Client Application at this time. Please try again at another time."
        }
    }
}
```

**Authentication**
Basic Authentication using Login ID and Password

**Authorization**
Any active Hybrid Data Pipeline user

**Get registered application by ID**

**Purpose**
Returns registered application by ID.
URL
https://<myserver>:<port>/api/mgmt/oauth/client/applications/{id}

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application.</td>
<td>It must be a valid application ID.</td>
</tr>
</tbody>
</table>

Response Definition
The response takes the following format. The parameters of the response are described in the table that follows.

```
{
    "id":"app_id,
    "name": "app_name"
    "description": "app_description"
    "redirectUrls": [
        "redirect_url1","redirect_url2",...
    ],
    "clientId": "string",
    "clientSecret": "string"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application. This ID is used for tracking applications in Hybrid Data Pipeline.</td>
<td>An auto-generated application ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the application</td>
<td>A string with a maximum length of 128 characters.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>User specified description of the application.</td>
<td>A string with a maximum length of 256 characters</td>
</tr>
<tr>
<td>&quot;redirectUrls&quot;</td>
<td>List of authorized URLs</td>
<td>This may be one valid URL or a comma separated list of valid URLs.</td>
</tr>
</tbody>
</table>
Valid Values Description Property
"clientId" The client ID is generated when the client application is registered. This ID is required when client applications initiate OAuth authorization. An auto-generated value used when client applications initiate OAuth authorization.
"clientSecret" The client secret is generated when the client application is registered. This secret is required when client applications initiate OAuth authorization. An auto-generated value used when client applications initiate OAuth authorization.

Sample Server Success Response

Status code: 200
Successful response

```json
{
    "id":19,
    "name":"Application1",
    "description":"Application1 for Create with all Fields",
    "redirectUrls":["bedford.progresstest.com","americas.progresstest.com"],
    "clientId":"315368974.apps.hdptest.com",
    "clientSecret":"96dab351-cd80-4dfc-8756-8afe9896e92f"
}
```

Sample Server Failure Response

```json
{
    "error":{
        "code":222206630,
        "message":{
            "lang":"en-US",
            "value":"There is no OAuth Client Application with that id:{id}.."
        }
    }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

Any active Hybrid Data Pipeline user

Update registered application by ID

Purpose

Updates the registered application by ID.
**URL**

https://<myserver>;<port>/api/mgmt/oauth/client/applications/{id}

**Method**

PUT

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application.</td>
<td>It must be a valid application ID.</td>
</tr>
</tbody>
</table>

**Request Payload Parameters**

The request payload is a JSON object with the following format:

```json
{
    "name": "app_name",
    "description": "app_description",
    "redirectUrls": ["redirect_url1","redirect_url2",... ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application. This ID is used for tracking applications in Hybrid Data Pipeline.</td>
<td>An auto-generated application ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the application</td>
<td>A string with a maximum length of 128 characters.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>User specified description of the application.</td>
<td>A string with a maximum length of 256 characters</td>
</tr>
<tr>
<td>&quot;redirectUrls&quot;</td>
<td>List of authorized URLs</td>
<td>This may be one valid URL or a comma separated list of valid URLs.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Valid Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The client ID is generated when the client application is registered. This ID is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The client secret is generated when the client application is registered. This secret is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
</tbody>
</table>

**Sample Payload**

```json
{
    "id": 22,
    "name": "Application3 for Update",
    "description": "Description of Application will all fields",
    "redirectUrls": [
        "test.sforcestest.com",
        "mor.progresstest.com"
    ]
}
```

**Sample Server Success Response**

Status code: 200
OK

```json
{
    "id": 22,
    "name": "Application3 for Update",
    "description": "Description of Application will all fields",
    "redirectUrls": [
        "test.sforcestest.com",
        "mor.progresstest.com"
    ]
}
```

**Sample Server Failure Response**

```json
{
    "error":{
        "code":222206632,
        "message":{
            "lang":"en-US",
            "value":"Problem updating OAuth Client Application at this time. Please try again at another time."
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password
Authorization
Any active Hybrid Data Pipeline user

Delete registered application by ID

Purpose
Deletes OAuth Registered application by ID.

URL
https://<myserver>:<port>/api/mgmt/oauth/client/applications/{id}

Method
DELETE

URL Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application.</td>
<td>It must be a valid application ID.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successfully deleted third party app

Authentication
Basic Authentication using Login ID and Password

Authorization
Any active Hybrid Data Pipeline user

Reset client secret of registered application

Purpose
Sets the client secret of the specified application. This will result in the revoking of all access granted to that application.
URL

https://<myserver>:<port>/api/mgmt/oauth/client/applications/{id}/reset

Method

PUT

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application.</td>
<td>It should be a valid application ID.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

{
  "id":19,
  "name":"Application1",
  "description":"Application1 for Create with all Fields",
  "redirectUrls":["bedford.progresstest.com","americas.progresstest.com"],
  "clientId":"315368974.apps.hdptest.com",
  "clientSecret":"69dab351-cd80-4dfc-8756-8afe9896e92f"
}

Sample Server Failure Response

{
  "error":{
    "code":222206634,
    "message":{
      "lang":"en-US",
      "value":"Problem resetting OAuth Client secret at this time. Please try again at another time."
    }
  }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

Any active Hybrid Data Pipeline user
Get list of applications for which logged-in user has access

Purpose
Returns the list of applications for which the logged-in user has been granted access.

URL
https://<myserver>:<port>/api/mgmt/oauth/client/allowedapplications

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition
The response takes the following format. The parameters of the response are described in the table that follows.

```json
{
    "applications": [
    {
        "id": app_id,
        "name": app_name,
        "scopes": [
            "string"
        ]
    },
    ..
    ..
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID</td>
<td>An auto-generated application ID</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the application</td>
<td>A string with a maximum length of 128</td>
</tr>
<tr>
<td>&quot;scopes&quot;</td>
<td>An OAuth 2.0 scope specifies the resources that can be accessed by client applications.</td>
<td>Currently, the only supported scope is &quot;api.access.odata&quot;.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response
{  
  "applications": [  
    {  
      "id": 1,  
      "name": "TestOAuthApplication_1",  
      "scopes": [  
        "api.access.odata"  
      ]  
    },  
    {  
      "id": 3,  
      "name": "TestOAuthApplication_2",  
      "scopes": [  
        "api.access.odata"  
      ]  
    }  
  ]  
}

Sample Server Failure Response

{  
  "error":{  
    "code":222206631,  
    "message":{  
      "lang":"en-US",  
      "value":"Problem getting OAuth Client Application at this time. Please try again at another time."  
    }  
  }  
}

Authentication
Basic Authentication using Login ID and Password

Authorization
Any active Hybrid Data Pipeline user

Revoke access granted for the given application ID

Purpose
Revokes the access granted for the given application ID.

URL
https://<myserver>:<port>/api/mgmt/oauth/client/allowedapplications/{id}

Method
DELETE
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The application ID is automatically generated with the successful registration of the application.</td>
<td>It must be a valid application ID.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successfully revoked access for third party app

Sample Server Failure Response

```json
{
   "error":{
      "code":222206633,
      "message":{
         "lang":"en-US",
         "value":"Problem deleting OAuth Client Application at this time. Please try again at another time."
      }
   }
}
```

Authentication

Basic Authentication using Login ID and Password

Authorization

Any active Hybrid Data Pipeline user

Generate access token and refresh token

Purpose

Generates access token and refresh token, using either of the two grant types- password grant type or refresh_token grant type.

URL

https://<myserver>:<port>/oauth2/token

Method

POST
**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

**Payload Parameters**

When the grant type is refresh_token, the following payload parameters are required:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;grant_type&quot;</td>
<td>The grant type used for OAuth flow</td>
<td>This can be either password or refresh_token. In this case, it is refresh_token.</td>
</tr>
<tr>
<td>&quot;refresh_token&quot;</td>
<td>The refresh token that had been issued to the application.</td>
<td>A valid refresh token issued by Hybrid Data Pipeline.</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The client ID is generated when the client application is registered. This ID is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The client secret is generated when the client application is registered. This secret is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
</tbody>
</table>

When the grant type is password, the following payload parameters are required:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;grant_type&quot;</td>
<td>The grant type used for OAuth flow</td>
<td>This can be either password or refresh_token. In this case, it is password.</td>
</tr>
<tr>
<td>&quot;scope&quot;</td>
<td>An OAuth 2.0 scope specifies the resources that can be accessed by client applications.</td>
<td>Currently, the only supported scope is api.access.odata.</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The client ID is generated when the client application is registered. This ID is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The client secret is generated when the client application is registered. This secret is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
</tbody>
</table>
Valid Values

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;username&quot;</td>
<td>User credentials</td>
<td>String</td>
</tr>
<tr>
<td>&quot;password&quot;</td>
<td>User Credentials</td>
<td>String</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 200
Successful response

{
  "access_token": "string",
  "refresh_token": "string",
  "expires_in": "string"
}

Sample Server Failure Response

{
  "error":{
    "code":222206628,
    "message":{
      "lang":"en-US",
      "value":"Problem creating OAuth Client Application at this time. Please try again at another time."
    }
  }
}

Authentication

Basic Authentication using Login ID and Password

Authorization

Any active Hybrid Data Pipeline user

Authorize token

Purpose

Before the user reaches authorize end-point, Hybrid Data Pipeline validates whether the user is logged in or not. In case the user is not logged in, he/she is redirected to the login page. After logging in, the user is redirected to the specified url. The endpoint then validates the client id and redirect url and the user will be presented with consent screen. The user can give consent by clicking on the allow button. After the user gives the consent, an auth code is generated and sent to the redirect url. The client application will then exchange that authcode for access and refresh tokens.

URL

https://<myserver>:<port>/oauth2/authorize
**Method**

POST

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;scope&quot;</td>
<td>Scopes are used to grant an application different levels of access to data on behalf of the end user.</td>
<td>Currently, the only supported scope is &quot;api.access.odata&quot;.</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The client ID is generated when the client application is registered. This ID is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The client secret is generated when the client application is registered. This secret is required when client applications initiate OAuth authorization.</td>
<td>An auto-generated value used when client applications initiate OAuth authorization.</td>
</tr>
<tr>
<td>&quot;response_type&quot;</td>
<td>The grant type being used.</td>
<td>The response type must be 'code'</td>
</tr>
<tr>
<td>&quot;redirect_uri&quot;</td>
<td>List of authorized URLs</td>
<td>This may be one valid URL or a comma separated list of valid URLs.</td>
</tr>
</tbody>
</table>

**Response Definition**

```json
{
  "access_token": "string",
  "refresh_token": "string",
  "expires_in": "string"
}
```

**Sample Server Success Response**

Status code: 200
Created

```json
{
  "access_token": "fdb8fdebced1d03ce5e612ng",
  "refresh_token": "u67rkt4drt5ieigfd0bce58f",
  "expires_in": "600"
}
```
Sample Server Failure Response

```json
{
   "error":{
      "code":222206628,
      "message":{
         "lang":"en-US",
         "value":"Problem creating OAuth Client Application at this time. Please try again at another time."
      }
   }
}
```

**Authentication**

Basic Authentication using Login ID and Password

**Authorization**

Any active Hybrid Data Pipeline user

**OAuth API for Google Analytics connectivity**

The OAuth API for Google Analytics allows for integration of Hybrid Data Pipeline with a Google Analytics OAuth 2.0 authorization flow. The OAuth API for Google Analytics is comprised of the OAuth applications API and the OAuth profiles API.

The OAuth applications API allows Hybrid Data Pipeline to identify itself as a registered Google Analytics application with the creation of an OAuth application object. The OAuth application object holds the OAuth client ID and secret. The permissions required to use the OAuth applications API depend on the operation being used and the tenant environment. See OAuth applications API for details.

The OAuth profiles API permits Hybrid Data Pipeline access to Google Analytics through the creation of an OAuth profile object. To complete OAuth authorization, the OAuth profile object provides OAuth refresh and access tokens to Google Analytics. OAuth profiles are created or selected for data sources, and a single OAuth profile can be used for multiple data sources on a Google Analytics data store. Since OAuth profiles are associated with data sources, a user must have a corresponding data source permission to create, view, modify, or delete OAuth profiles. For example, to create an OAuth profile, a user must have the CreateDataSource (1) permission. See OAuth profiles API for details.

See the following topics for more information.

- OAuth applications API
- OAuth profiles API

**OAuth applications API**

The OAuth applications API allows Hybrid Data Pipeline to identify itself as a registered Google Analytics application with the creation of an OAuth application object. The OAuth application object holds the OAuth client_id and client_secret. The permissions required to use the OAuth applications API depend on the operation being used and the tenant environment.
In a multitenant environment, an OAuth application object must belong to a particular tenant, and only one OAuth application object can be created for a Google Analytics data store for each tenant. When an OAuth application object exists in the system tenant and another exists in a child tenant, the OAuth application object in the child tenant will override the one in the system tenant for the users who belong to the child tenant. How the OAuth applications API may be used depends, in part, on the permissions the administrator has. With the Administrator (12) permission, a user can create an OAuth application object in any tenant across the system. With the MgmtAPI (11) and OAuth (28) permissions, a user in the system tenant can create an OAuth application object for the system tenant. This user can also create OAuth application objects for tenants for which he or she has administrative access. With the MgmtAPI (11) and OAuth (28) permissions, a user in a child tenant can create an OAuth application object only in the tenant in which he or she resides.

The following table lists the operations that can be performed with the OAuth applications API.

<table>
<thead>
<tr>
<th>Task</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve OAuth applications</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthapps</td>
</tr>
<tr>
<td>Create an OAuth application object</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthapps</td>
</tr>
<tr>
<td>Retrieve an OAuth application object</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthapps/{id}</td>
</tr>
<tr>
<td>Update an OAuth application object</td>
<td>PUT</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthapps/{id}</td>
</tr>
<tr>
<td>Delete an OAuth application object</td>
<td>DELETE</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthapps/{id}</td>
</tr>
</tbody>
</table>

**Get OAuth applications**

**Purpose**

Retrieves a list of OAuth application objects. OAuth application objects contain the OAuth client ID and secret.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**

https://<myserver>:<port>/api/mgmt/oauthapps

**Filter by a query parameter**

A user can also filter query results by tenant by appending the URL with a ?tenantId=<tenant_id> or ?tenantName=<tenant_name> query parameter. For example:

https://<myserver>:<port>/api/mgmt/oauthapps?tenantId=<tenant_id>

**Method**

GET

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
Response Definition

The response takes the following format.

```json
{
  "applications": [
    {
      "id": "oauth_application_id",
      "name": "oauth_application_name",
      "dataStore": "data_store_id",
      "tenantId": "tenant_id",
      "description": "oauth_application_description"
    },
    ...,
  ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the OAuth application object.</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth application object.</td>
<td>The user-specified name of the OAuth application object. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store for which the OAuth application object is being created.</td>
<td>The only data store which Hybrid Data Pipeline currently supports access to is Google Analytics. Therefore, the only valid value is the Google Analytics data store ID: 54.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the OAuth application and data store belong.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth application object.</td>
<td>A description provided by the user.</td>
</tr>
</tbody>
</table>

Sample Server Response

**Status code: 200**

**Successful response**

```json
{
  "applications": [
    {
      "id": "11",
      "name": "HDP system OAuth app",
      "dataStore": "54",
      "tenantId": 1,
      "description": "Hybrid Data Pipeline OAuth application object for Google Analytics"
    },
    {
      "id": "17",
      "name": "TenantA OAuth app",
      "dataStore": "54",
      "tenantId": 303,
      "description": "TenantA OAuth application object for another data store"
    }
  ]
}
```
Authentication
Basic Authentication using Login ID and Password

Authorization
Permissions apply in the following manner.

- With the Administrator (12) permission, a user can view all OAuth application objects across the system.
- With the MgmtAPI (11) and OAuth (28) permissions, a user in the system tenant can view existing OAuth application objects in the system tenant and in any tenants for which he or she has administrative access.
- With the MgmtAPI (11) and OAuth (28) permissions, a user in a child tenant can only view OAuth application objects in the tenant in which he or she resides.

Create an OAuth application object

Purpose
Creates an OAuth application object that holds the OAuth client ID and secret. An OAuth application ID is automatically generated. This ID is used to link an OAuth application with an OAuth profile.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/oauthapps

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Definition
The request takes the following format.

```json
{
   "name": "oauth_application_name",
   "dataStore": data_store_id,
   "tenantId": tenant_id,
   "description": "oauth_application_description",
   "clientId": "client_id",
}
```
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth application object.</td>
<td>Required</td>
<td>The user-specified name of the OAuth application object. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store for which the OAuth application object is being created.</td>
<td>Required</td>
<td>The only data store which Hybrid Data Pipeline currently supports access to is Google Analytics. Therefore, the only valid value is the Google Analytics data store ID: 54.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the OAuth application and data store belong.</td>
<td>Optional</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth application object.</td>
<td>Optional</td>
<td>A description provided by the user.</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The OAuth client_id generated by Google when an application is registered with the Analytics API in the Google Developer Console.</td>
<td>Required</td>
<td>A valid client_id.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The OAuth client_secret generated by Google when an application is registered with the Analytics API in the Google Developer Console.</td>
<td>Required</td>
<td>A valid client_secret.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```json
{
   "name": "TenantA OAuth app",
   "dataStore": 54,
   "tenantId": 303,
   "description": "TenantA OAuth application object for Google Analytics",
   "clientId": "asdfjasdljfasdkjf",
   "clientSecret": "1912308409123890"
}
```
Sample Server Response

Status code: 201
Successful response

{
   "id": "17",
   "name": "TenantA OAuth app",
   "dataStore": 54,
   "tenantId": 303,
   "description": "TenantA OAuth application object for Google Analytics",
   "clientId": "asdfjasdljfasdkjf",
   "clientSecret": "1912308409123890"
}

Authentication

Basic Authentication using Login ID and Password.

Authorization

Permissions apply in the following manner.

• With the Administrator (12) permission, a user can create an OAuth application object in any tenant across the system.

• With the MgmtAPI (11) and OAuth (28) permissions, a user in the system tenant can create an OAuth application object for the system tenant. This user can also create OAuth application objects for tenants for which he or she has administrative access.

• With the MgmtAPI (11) and OAuth (28) permissions, a user in a child tenant can create an OAuth application object only in the tenant in which he or she resides.

Get an OAuth application object

Purpose

Retrieves an OAuth application object.

---

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

---

URL

https://<myserver>:<port>/api/mgmt/oauthapps/{id}

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
The URL parameter `{id}` is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{id}</code></td>
<td>The ID of the OAuth application object.</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
</tbody>
</table>

**Response Definition**

The response takes the following format.

```json
{
  "id": "oauth_application_id",
  "name": "oauth_application_name",
  "dataStore": data_store_id,
  "tenantId": tenant_id,
  "description": "oauth_application_description",
  "clientId": "client_id",
  "clientSecret": "client_secret"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the OAuth application object.</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuthApplication.</td>
<td>The user-specified name of the OAuth application object. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store for which the OAuth application object is being created.</td>
<td>The only data store which Hybrid Data Pipeline currently supports access to is Google Analytics. Therefore, the only valid value is the Google Analytics data store ID: 54.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the OAuth application and data store belong. When a tenant ID is not specified, the OAuth application is created for the tenant to which the user belongs.</td>
<td>A valid tenant ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth application object.</td>
<td>A description provided by the user.</td>
</tr>
</tbody>
</table>
### Sample Response Payload

Status code: 200
Successful response

```
{
    "id": "17",
    "name": "TenantA OAuth app",
    "dataStore": 54,
    "tenantId": 303,
    "description": "TenantA OAuth application object for Google Analytics",
    "clientId": "asdfjasdljfasdkjfl",
    "clientSecret": "1912308409123890"
}
```

### Authentication

Basic Authentication using Login ID and Password.

### Authorization

Permissions apply in the following manner.

- With the Administrator (12) permission, a user can view any OAuth application object across the system.
- With the MgmtAPI (11) and OAuth (28) permissions, a user in the system tenant can view an OAuth application object in the system tenant and in any tenants for which he or she has administrative access.
- With the MgmtAPI (11) and OAuth (28) permissions, a user in a child tenant can only view an OAuth application object in the tenant in which he or she resides.

### Update an OAuth application object

#### Purpose

Updates an OAuth application object.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

#### URL

https://<myserver>:<port>/api/mgmt/oauthapps/{id}

#### Method

PUT
**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the OAuth application object.</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
</tbody>
</table>

**Request Payload Parameters**

The request takes the following format.

```
{
    "name": "oauth_application_name",
    "dataStore": data_store_id,
    "tenantId": tenant_id,
    "description": "oauth_application_description",
    "clientId": "client_id",
    "clientSecret": "client_secret"
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth application object.</td>
<td>Required</td>
<td>The user-specified name of the OAuth application object. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;dataStore&quot;</td>
<td>The ID of the data store for which the OAuth application object is being created.</td>
<td>Required</td>
<td>The only data store which Hybrid Data Pipeline currently supports access to is Google Analytics. Therefore, the only valid value is the Google Analytics data store ID: 54.</td>
</tr>
<tr>
<td>&quot;tenantId&quot;</td>
<td>The ID of the tenant to which the OAuth application and data store belong. When a tenant ID is not specified, the OAuth application is created for the tenant to which the user belongs.</td>
<td>Optional</td>
<td>A valid tenant ID.</td>
</tr>
</tbody>
</table>
### Property Table

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth application object.</td>
<td>Optional</td>
<td>A description provided by the user.</td>
</tr>
<tr>
<td>&quot;clientId&quot;</td>
<td>The OAuth client ID generated by Google when an application is registered with the Analytics API in the Google Developer Console.</td>
<td>Required</td>
<td>A valid client ID.</td>
</tr>
<tr>
<td>&quot;clientSecret&quot;</td>
<td>The OAuth client secret generated by Google when an application is registered with the Analytics API in the Google Developer Console.</td>
<td>Required</td>
<td>A valid client secret.</td>
</tr>
</tbody>
</table>

### Sample Request Payload

```json
{
   "name": "TenantB OAuth app",
   "dataStore": 54,
   "tenantId": 623,
   "description": "TenantB OAuth application object for Google Analytics",
   "clientId": "asdfjasdljfasdkjf",
   "clientSecret": "1912308409123890"
}
```

### Sample Server Response

```
Status code: 200
Successful response

{
   "id": "93",
   "name": "TenantB OAuth app",
   "dataStore": 54,
   "tenantId": 623,
   "description": "TenantB OAuth application object for Google Analytics",
   "clientId": "asdfjasdljfasdkjf",
   "clientSecret": "1912308409123890"
}
```

### Authentication

Basic Authentication using Login ID and Password.

### Authorization

Permissions apply in the following manner.

- With the Administrator (12) permission, a user can modify any OAuth application object across the system.
- With the MgmtAPI (11) and OAuth (28) permissions, a user in the system tenant can modify an OAuth application object in the system tenant and in any tenants for which he or she has administrative access.
- With the MgmtAPI (11) and OAuth (28) permissions, a user in a child tenant can only modify an OAuth application object in the tenant in which he or she resides.
Delete an OAuth application object

Purpose
Deletes an OAuth application object. Deleting an OAuth application object deletes any associated OAuth profile objects.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/oauthapps/{id}

Method
DELETE

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} described is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the OAuth application object.</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
</tbody>
</table>

Sample Server Success Response

Status code: 204
Successful response

```json
{
    "success":true
}
```

Sample Server Failure Response

```json
{
    "error": {
        "code": "222207711",
        "message": {
            "lang": "en-US",
            "value": "Invalid OAuthApplicationId: 223344"
        }
    }
}
```
Authentication
Basic Authentication using Login ID and Password.

Authorization
Permissions apply in the following manner.

• With the Administrator (12) permission, a user can delete any OAuth application object across the system.
• With the MgmtAPI (11) and OAuth (28) permissions, a user in the system tenant can delete an OAuth application object in the system tenant and in any tenants for which he or she has administrative access.
• With the MgmtAPI (11) and OAuth (28) permissions, a user in a child tenant can only delete an OAuth application object in the tenant in which he or she resides.

OAuth profiles API
The OAuth profiles API permits Hybrid Data Pipeline access to Google Analytics through the creation of an OAuth profile object. To complete OAuth authorization, the OAuth profile object provides OAuth refresh and access tokens to Google Analytics. OAuth profiles are created or selected for data sources, and a single OAuth profile can be used for multiple data sources on a Google Analytics data store. Since OAuth profiles are associated with data sources, a user must have a corresponding data source permission to create, view, modify, or delete OAuth profiles. For example, to create an OAuth profile, a user must have the CreateDataSource (1) permission.

The following table lists the operations that can be performed with the OAuth profiles API.

<table>
<thead>
<tr>
<th>Task</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve OAuth profiles</td>
<td>GET</td>
<td><code>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthprofiles</code></td>
</tr>
<tr>
<td>Create an OAuth profile</td>
<td>POST</td>
<td><code>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthprofiles</code></td>
</tr>
<tr>
<td>Retrieve an OAuth profile</td>
<td>GET</td>
<td><code>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthprofiles/{id}</code></td>
</tr>
<tr>
<td>Update an OAuth profile</td>
<td>PUT</td>
<td><code>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthprofiles/{id}</code></td>
</tr>
<tr>
<td>Delete an OAuth profile</td>
<td>DELETE</td>
<td><code>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthprofiles/{id}</code></td>
</tr>
<tr>
<td>Retrieve statistics for an OAuth profile</td>
<td>GET</td>
<td><code>https://&lt;myserver&gt;:&lt;port&gt;/api/mgmt/oauthprofiles/{id}/stats</code></td>
</tr>
</tbody>
</table>

Get OAuth profiles

Purpose
Retrieves a list of OAuth profiles available to the user.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
`https://<myserver>:<port>/api/mgmt/oauthprofiles`
Method
GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Response Definition

The response takes the following format.

```
{
   "profiles": [
   {
      "id": "oauth_profile_id",
      "name": "oauth_profile_id",
      "description": "oauth_profile_description"
   },
   ...
   ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>The ID of the OAuth profile.</td>
<td>The automatically generated OAuth profile ID.</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth profile.</td>
<td>The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth profile.</td>
<td>A description provided by the user.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 200
Successful response

```
{
   "profiles": [
   {
      "id": "33",
      "name": "Google User_1",
      "description": "OAuth profile 1"
   },
   {
      "id": "39",
      "name": "Google User_2",
      "description": "OAuth profile 2"
   }
   ]
}
```

Authentication

Basic Authentication using Login ID and Password
Authorization
The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

Create an OAuth profile

Purpose
Creates an OAuth profile that can be associated with a data source for access to Google Analytics.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/oauthprofiles

Method
POST

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Request Payload Parameters
The request takes the following format.

```
{
  "name": "oauth_profile_name",
  "oauthAppId": "oauth_application_id",
  "description": "oauth_profile_description",
  "accessToken": "access_token",
  "refreshToken": "refresh_token"
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth profile.</td>
<td>Required</td>
<td>The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;oauthAppId&quot;</td>
<td>The ID of the OAuth application object.</td>
<td>Required</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth profile.</td>
<td>Optional</td>
<td>A description provided by the user.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Usage</td>
<td>Valid Values</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>&quot;accessToken&quot;</td>
<td>The access token includes the credential information required to gain access to the Google Analytics API.</td>
<td>Optional</td>
<td>A valid access token.</td>
</tr>
<tr>
<td>&quot;refreshToken&quot;</td>
<td>The refresh token is used to generate new access tokens.</td>
<td>Required</td>
<td>A valid refresh token.</td>
</tr>
</tbody>
</table>

**Sample Request Payload**

```json
{
    "name": "Google_User_1",
    "oauthAppId": 17,
    "description": "OAuth profile 1",
    "accessToken": "111c334445e55",
    "refreshToken": "222d88899966fa"
}
```

**Sample Server Success Response**

```
Status code: 201
Successful response

{
    "id": 33,
    "name": "Google_User_1",
    "oauthAppId": 17,
    "description": "OAuth profile 1",
    "accessToken": "111c334445e55",
    "refreshToken": "222d88899966fa"
}
```

**Sample Server Failure Response**

```
{
    "error": {
        "code": "222207710",
        "message": {
            "lang": "en-US",
            "value": "Invalid OAuthProfileId: 1"
        }
    }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and CreateDataSource (1) permissions.

**Get an OAuth profile**

**Purpose**

Retrieves information on an OAuth profile.
**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**URL**
https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}

**Method**
GET

**URL parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the OAuth profile.</td>
<td>The automatically generated OAuth profile ID.</td>
</tr>
</tbody>
</table>

**Response Definition**
The response takes the following format.

```json
{
  "name": "oauth_profile_name",
  "oauthAppId": oauth_application_id,
  "description": "oauth_profile_description",
  "accessToken": "access_token",
  "refreshToken": "refresh_token"
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth profile.</td>
<td>The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;oauthAppId&quot;</td>
<td>The ID of the OAuth application object.</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth profile.</td>
<td>A description provided by the user.</td>
</tr>
<tr>
<td>&quot;accessToken&quot;</td>
<td>The access token includes the credential information required to gain access to the Google Analytics API.</td>
<td>A valid access token.</td>
</tr>
<tr>
<td>&quot;refreshToken&quot;</td>
<td>The refresh token is used to generate new access tokens.</td>
<td>A valid refresh token.</td>
</tr>
</tbody>
</table>
Sample Response Payload

Status code: 200
Successful response

```
{
  "id": 33,
  "name": "Google_User_1",
  "oauthAppId": 17,
  "description": "OAuth profile 1",
  "accessToken": "111c334445e55",
  "refreshToken": "222d88899966fa"
}
```

Authentication

Basic Authentication using Login ID and Password.

Authorization

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

Update an OAuth profile

Purpose

Updates an OAuth profile. Users can only edit OAuth profiles they own or have created.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL

https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}

Method

PUT

URL parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the OAuth profile.</td>
<td>The automatically generated OAuth profile ID.</td>
</tr>
</tbody>
</table>
Request Payload Definition
The response takes the following format.

```json
{
  "name": "oauth_profile_name",
  "oauthAppId": oauth_application_id,
  "description": "oauth_profile_description",
  "accessToken": "access_token",
  "refreshToken": "refresh_token"
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the OAuth profile.</td>
<td>Required</td>
<td>The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;oauthAppId&quot;</td>
<td>The ID of the OAuth application object.</td>
<td>Required</td>
<td>The automatically generated OAuth application ID.</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>A description of the OAuth profile.</td>
<td>Optional</td>
<td>A description provided by the user.</td>
</tr>
<tr>
<td>&quot;accessToken&quot;</td>
<td>The access token includes the credential information required to gain access to the Google Analytics API.</td>
<td>Optional</td>
<td>A valid access token.</td>
</tr>
<tr>
<td>&quot;refreshToken&quot;</td>
<td>The refresh token is used to generate new access tokens.</td>
<td>Required</td>
<td>A valid refresh token.</td>
</tr>
</tbody>
</table>

Sample Request Payload

```json
{
  "name": "Google_User_1",
  "oauthAppId": 17,
  "description": "OAuth profile 1",
  "accessToken": "111c334445e55",
  "refreshToken": "222d88899966fa"
}
```

Sample Server Response

Status code: 200
Successful response

```json
{
  "id": 33,
  "name": "Google_User_1",
  "oauthAppId": 17,
  "description": "OAuth profile 1",
  "accessToken": "111c334445e55",
  "refreshToken": "222d88899966fa"
}
```
Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and ModifyDataSource (3) permissions.

Delete the specified OAuthProfile

Purpose
Deletes an OAuth profile. Deleting an OAuth profile removes its references from data sources.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}

URL parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the OAuth profile.</td>
<td>The automatically generated OAuth profile ID.</td>
</tr>
</tbody>
</table>

Method
DELETE

Sample Server Success Response

Status code: 204
Successful response

```
{
  "success":true
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": "222207710",
```

"message": {
  "lang": "en-US",
  "value": "Invalid OAuthProfileId: 30"
}

Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and DeleteDataSource (4) permissions.

Get statistics for an OAuth profile

Purpose
Retrieves statistics for an OAuth profile. Statistics include the number of data sources that use the OAuth profile and information on the data sources themselves.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL
https://<myserver>:<port>/api/mgmt/oauthprofiles/{id}/stats

Method
GET

URL parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

The URL parameter {id} is required.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>{id}</td>
<td>The ID of the OAuth profile.</td>
<td>The automatically generated OAuth profile ID.</td>
</tr>
</tbody>
</table>

Response definition
The response takes the following format.

```json
{
  "dataSourcesLinked": integer,
  "dataStoreId": data_store_id,
}
"dataSources": [  
  {  
    "id": data_source_id,
    "name": data_source_name,
    "description": data_source_description
  },
  ...
]}

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;dataSourcesLinked&quot;</td>
<td>The number of data sources which the OAuth profile links to.</td>
<td>A non-negative integer.</td>
</tr>
<tr>
<td>&quot;dataStoreId&quot;</td>
<td>The ID of the data store for which the OAuth application object is being created.</td>
<td>The only data store which Hybrid Data Pipeline currently supports access to is Google Analytics. Therefore, the only valid value is the Google Analytics data store ID: 54.</td>
</tr>
<tr>
<td>&quot;dataSources&quot;</td>
<td>A list of the data sources linked to the OAuth profile.</td>
<td>A comma separated list of data source objects that contains the id, name, and description for each linked data source.</td>
</tr>
</tbody>
</table>

Sample Server Response

Status code: 200  
Successful response

{
  "dataSourcesLinked": 2,
  "dataStoreId": 54,
  "dataSources": [  
    {  
      "id": "503",
      "name": "GAtest",
      "description": "test"
    },
    {  
      "id": "611",
      "name": "GApred",
      "description": "production"
    }
  ]
}

Sample server failure response

{
  "error": {
    "code": "222207710",
    "message": {
      "lang": "en-US",
      "value": "Invalid OAuthProfileId: 30"
    }
  }
}
**Authentication**
Basic Authentication using Login ID and Password.

**Authorization**
The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

**Schema API**
The Schema API is an extension of the Data Sources API. The Schema API can be used to retrieve the information needed to configure a schema for OData connectivity.

*Important:* For backend data stores that support schemas, Hybrid Data Pipeline provides an option to restrict the metadata exposed by the service to a single schema. When a schema has been specified for the **Metadata Exposed Schemas** option in the Web UI (or the `HDPMetadataExposedSchemas` property via the Data Sources API), the Schema API can only be used to query the specified schema. For details on **Metadata Exposed Schemas**, see the parameters topic for your data source type.

The following table lists the operations that can be performed and their associated URLs. A detailed description for these operations follows.

<table>
<thead>
<tr>
<th>Task</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve a list of available schemas</td>
<td>GET</td>
<td><code>&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceid}/schemas</code></td>
</tr>
<tr>
<td>Retrieve table names</td>
<td>GET</td>
<td><code>&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceid}/schemas/</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;schemaName&gt;/tables</code></td>
</tr>
<tr>
<td>Retrieve table information</td>
<td>GET</td>
<td><code>&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceid}/schemas/</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;schemaName&gt;/tables/&lt;tableName&gt;</code></td>
</tr>
<tr>
<td>Retrieve column information for a table</td>
<td>GET</td>
<td><code>&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceid}/schemas/</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;schemaName&gt;/tables/tableName/columns</code></td>
</tr>
<tr>
<td>Retrieve primary keys for a table</td>
<td>GET</td>
<td><code>&lt;myserver&gt;:&lt;port&gt;/api/mgmt/datasources/{datasourceid}/schemas/</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>&lt;schemaName&gt;/tables/&lt;tableName&gt;/primarykeys</code></td>
</tr>
</tbody>
</table>

**Get schemas**

**Purpose**
Retrieves a list of the schemas for a particular data source.

*Important:* The schemas returned will be restricted to a single schema when a schema has been specified for the **Metadata Exposed Schemas** option in the Web UI (or the `HDPMetadataExposedSchemas` property via the DataSource API). For details, see the parameters topic for your data source type.
Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

URL

https://<myserver>:<port>/api/mgmt/datasources/<datasourceId>/schemas

Method

GET

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;datasourceId&quot;</td>
<td>The ID of the data source</td>
<td>Required</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>

Response definition

The response takes the following format. The properties of the response are described in the table that follows.

```
{
   "schemas": [
      {
         "name": "schema_name"
      },
      ...  
   ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>For data stores that support schemas, the name of a schema associated with the data source. For data stores that do not support schemas, a dash (−) is returned.</td>
<td>Required</td>
<td>For data stores that support schemas, a valid schema name. For data stores that do not support schemas, a dash (−) is returned.</td>
</tr>
</tbody>
</table>

Sample server success response

Data stores that support schemas, return a payload similar to the following.

```
{
   "schemas": [
      
```

Data stores that do not support schemas, return a payload similar to the following.

```
{
  "schemas": [
    {
      "name": "-
    }
  ]
}
```

**Sample server failure response**

```
{
  "error": {
    "code": "222207011",
    "message": {
      "lang": "en-US",
      "value": "Invalid DataSource ID {0}"}
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

## Get table names

**Purpose**

Retrieves the names of the tables associated with a data source through the specified schema. For data stores that do not support schemas, retrieves the names of all tables associated with the data source.

**Important:** When a schema has been specified for the **Metadata Exposed Schemas** option in the Web UI (or the **HDPMetadataExposedSchemas** property via the Data Sources API), the Schema API can only be used to query the specified schema. If the schema specified for **Metadata Exposed Schemas** does not match the schema in the Schema API URL, then an empty result set will be returned. For details on **Metadata Exposed Schemas**, see the parameters topic for your data source type.

**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also **Managing resources on behalf of users** on page 1281.
URL

For data stores that support schemas

https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables

For data stores that do not support schemas

For data stores that do not support schemas, use a dash (−) as the identifier in the URL when retrieving information about tables. For example:

https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>-/tables

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;datasourceId&quot;</td>
<td>The ID of the data source</td>
<td>Required</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
</tbody>
</table>
| "schemaName"  | For data stores that support schemas, the name of a schema associated with the data source.  
For data stores that do not support schemas, a dash (−) is returned. | Required      | For data stores that support schemas, a valid schema name.  
For data stores that do not support schemas, a dash (−) is returned. |

Method

GET

Response definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
   "tables": [
   {
      "name": "tableName1"
   },
   {
      "name": "tableName2"
   }
}
```
Valid Values Usage Description Property

"name"  The names of the tables associated with the data source  Required  A table name can contain only alphanumeric characters and the underscore character.

**Sample server success response**

```json
{
  "tables": [
    {
      "name": "Account"
    },
    {
      "name": "Address"
    }
  ]
}
```

**Sample server failure response**

```json
{
  "error":{
    "code": 222207062,
    "message":{
      "lang":"en-US",
      "value":"The schema mySchemaName does not exist."
    }
  }
}
```

**Authentication**

Basic Authentication using Login ID and Password.

**Authorization**

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

**Get table information**

**Purpose**

Retrieves the details of the specified table.

**Important:** When a schema has been specified for the Metadata Exposed Schemas option in the Web UI (or the `HDPMetadataExposedSchemas` property via the Data Sources API), the Schema API can only be used to query the specified schema. If the schema specified for Metadata Exposed Schemas does not match the schema in the Schema API URL, then an empty result set will be returned. For details on Metadata Exposed Schemas, see the parameters topic for your data source type.
**Note:** An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

**Method**
GET

**URL**
For data stores that support schemas

https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables/<tableName>?user=<userName>

For data stores that do not support schemas

https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/-/tables/<tableName>?user=<userName>

**URL Parameters**

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;datasourceId&quot;</td>
<td>The ID of the data source</td>
<td>Required</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;schemaName&quot;</td>
<td>For data stores that support schemas, the name of a schema-associated with the data source. For data stores that do not support schemas, a dash (-) is returned.</td>
<td>Required</td>
<td>For data stores that support schemas, a valid schema name. For data stores that do not support schemas, a dash (-) is returned.</td>
</tr>
<tr>
<td>&quot;tableName&quot;</td>
<td>The name of the table for which information is being retrieved</td>
<td>Required</td>
<td>A table name can contain only alphanumeric characters and the underscore character.</td>
</tr>
</tbody>
</table>

**Response Definition**
The response takes the following format. The properties of the response are described in the table that follows.

```json
{
  "table": {
    "name": "tableName",
    "hasPrimaryKey": boolean,
    "columns": [ 
      { 
        "name": "colName1",
        "isPrimaryKey": Boolean
      },
    ]
  }
}
```
ValidValuesDescriptionProperty

Atablenamecancontainonlyalphanumericcharactersandthe underscorecharacter.

{| Property | Description | Valid Values |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the table for which information is being retrieved</td>
<td>A table name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;hasPrimaryKey&quot;</td>
<td>Specifies whether the table contains a primary key</td>
<td>true</td>
</tr>
<tr>
<td>If true, the table has a primary key. If false, the table does not have a primary key.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;columns&quot;</td>
<td>Provides a list of columns in the table. If the table has a primary key, this parameter identifies the column or columns that comprise the primary key.</td>
<td>A comma separated list of column objects</td>
</tr>
<tr>
<td>The name property specifies the column name. The isPrimaryKey property is a Boolean. If true, the column is a primary key or comprises the primary key. If false, the column is not a primary key and the property is not returned. Note that the schema might specify more than one column to define the primary key.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sample Server Success Response

```json
{
  "tables": {
    "name": "Account",
    "hasPrimaryKey": true,
    "columns": [
      {
        "name": "ROWID",
        "isPrimaryKey": true
      },
      {
        "name": "SYS_ISDELETED"
      },
      {
        "name": "MasterRecordId"
      },
      {
        "name": "SYS_NAME"
      }
    ]
  }
}
```
Sample Server Failure Response
Error 404 is returned if the schema does not exist.

Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

Get column information for a specified table

Purpose
Retrieve column information for a specified table.

Important: When a schema has been specified for the Metadata Exposed Schemas option in the Web UI (or the HDFMetadataExposedSchemas property via the Data Sources API), the Schema API can only be used to query the specified schema. If the schema specified for Metadata Exposed Schemas does not match the schema in the Schema API URL, then an empty result set will be returned. For details on Metadata Exposed Schemas, see the parameters topic for your data source type.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

Method
GET

URL
For data stores that support schemas

https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables/<tableName>/columns?user=<userName>

For data stores that do not support schemas

GET https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/-/tables/<tableName>/columns?user=<userName>

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
ValidValuesUsageDescriptionParameter

The ID is auto-generated when the datasource is created and cannot be changed.

"datasourceId"

Required

The ID of the data source

For data stores that support schemas, the name of a schema associated with the data source.
For data stores that do not support schemas, a dash (−) is returned.

"schemaName"

Required

For data stores that support schemas, a valid schema name.
For data stores that do not support schemas, a dash (−) is returned.

"tableName"

Required

The name of the table for which information is being retrieved

A table name can contain only alphanumeric characters and the underscore character.

"schemaName"

"tableName"

Method

GET

Response definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
  "columns": [
    {
      "name": "colname1",
      "isPrimaryKey": boolean
    },
    {
      "name": "colname2"
    }
  ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of the column for which information is being retrieved</td>
<td>A column name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>&quot;isPrimaryKey&quot;</td>
<td>Indicates whether the column is a primary key or comprises the primary key</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>If true, the column is a primary key or comprises the primary key.</td>
<td>If true, the column is a primary key or comprises the primary key.</td>
</tr>
<tr>
<td></td>
<td>If false, the column is not a primary key and the property is not returned.</td>
<td>If false, the column is not a primary key and the property is not returned.</td>
</tr>
<tr>
<td></td>
<td>Note that the schema might specify more than one column to define the primary key.</td>
<td>Note that the schema might specify more than one column to define the primary key.</td>
</tr>
</tbody>
</table>
Sample server success response

```
{
   "columns": [
   { "name": "ROWID" },
   { "name": "SYS_ISDELETED" },
   { "name": "MasterRecordId" },
   { "name": "SYS_NAME" }
   ]
}
```

Sample server failure response

Error 404 is returned if the schema does not exist.

Authentication

The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

Get primary key information for a specified table

Purpose

Retrieves the primary key for the specified table. If the table does not have a primary key assigned in the underlying data store, the schema may define a primary key consisting of one or more columns.

Important: When a schema has been specified for the Metadata Exposed Schemas option in the Web UI (or the HDFMetadataExposedSchemas property via the Data Sources API), the Schema API can only be used to query the specified schema. If the schema specified for Metadata Exposed Schemas does not match the schema in the Schema API URL, then an empty result set will be returned. For details on Metadata Exposed Schemas, see the parameters topic for your data source type.

Note: An administrator can execute this operation on behalf of a user by appending the user query parameter to the request and specifying a user name. See also Managing resources on behalf of users on page 1281.

Method

GET

URL

For data stores that support schemas

https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/<schemaName>/tables/<tableName>/primarykeys

For data stores that do not support schemas

https://<myserver>:<port>/api/mgmt/datasources/<datasourceid>/schemas/-/tables/<tableName>/primarykeys
URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Usage</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;datasourceld&quot;</td>
<td>The ID of the data source</td>
<td>Required</td>
<td>The ID is auto-generated when the data source is created and cannot be changed.</td>
</tr>
<tr>
<td>&quot;schemaName&quot;</td>
<td>For data stores that support schemas, the name of a schema associated with the data source. For data stores that do not support schemas, a dash (--) is returned.</td>
<td>Required</td>
<td>For data stores that support schemas, a valid schema name. For data stores that do not support schemas, a dash (--) is returned.</td>
</tr>
<tr>
<td>&quot;tableName&quot;</td>
<td>The name of the table for which information is being retrieved</td>
<td>Required</td>
<td>A table name can contain only alphanumeric characters and the underscore character.</td>
</tr>
</tbody>
</table>

Response definition

The response takes the following format. The properties of the response are described in the table that follows.

```json
{
    "primaryKeys": [
        {
            "name": "primaryKey1"
        },
        {
            "name": "MasterRecordId"
        }
    ]
}
```

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;name&quot;</td>
<td>The name of a primary key column, or a comma separated list of columns that comprise the primary key</td>
<td>A column name can contain only alphanumeric characters and the underscore character.</td>
</tr>
</tbody>
</table>

Sample server success response

The following response is returned for a table with a primary key comprised of two columns.

```json
{
    "primaryKeys": [
        {
            "name": "ROWID"
        },
        {
            "name": "MasterRecordId"
        }
    ]
}
```
Sample server failure response
Error 404 is returned if the schema does not exist.

Authentication
Basic Authentication using Login ID and Password.

Authorization
The user must have the MgmtAPI (11) and ViewDataSource (2) permissions.

Version Information API

Purpose
Retrieves version information, along with installation type details. The information returned includes the Product version, Data Access Service version and Web Application version.

URL
https://<myserver>:<port>/api/mgmt/version

Method
GET

URL Parameters
<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

Sample Server Success Response

{  
    "InstallType": "Licenced",  
    "HDPVersion": "4.3.0",  
    "WAPVersion": "4.3.0",  
    "DASVersion": "4.3.0"  
}

If you are using the evaluation version, the response will specify some additional details.

{  
    "InstallType": "Evaluation",  
    "EvalDaysRemaining": 89,  
    "EvalExpiryDate": "2018-05-06 T0 8:08:29.000Z",  
    "HDPVersion": "4.3.0",  
    "WAPVersion": "4.3.0",  
    "DASVersion": "4.3.0"  
}
Sample Server Failure Response

```json
{
   "error": {
      "code": 222208070,
      "message": {
         "lang": "en-US",
         "value": "Problem getting version information."
      }
   }
}
```

Authentication
Basic Authentication using Login ID and Password

Authorization
Any active Hybrid Data Pipeline user.

Password Policy API
The purpose of the Password Policy API is to access Password Policy enforced for the user and validate any password against the password policy.

You can perform the following operations with the Password Policy API.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Request</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Returns the password policy</td>
<td>GET</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/public/passwordpolicy</td>
</tr>
<tr>
<td>Validates the specified password</td>
<td>POST</td>
<td>https://&lt;myserver&gt;:&lt;port&gt;/api/public/passwordpolicy/validate</td>
</tr>
</tbody>
</table>

Get Password Policy

Purpose
Returns the details of the enforced Password Policy

URL
https://<myserver>:<port>/api/public/passwordpolicy

Method
GET
### URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;id&quot;</td>
<td>Password Policy id</td>
</tr>
<tr>
<td>&quot;name&quot;</td>
<td>Password Policy Name</td>
</tr>
<tr>
<td>&quot;description&quot;</td>
<td>Password Policy Description</td>
</tr>
<tr>
<td>&quot;policyDefinition&quot;</td>
<td>The definition of the password policy. Contains information about which rules are enforced in the password policy. Each rule has an associated ruleId, a rule title and a rule name. The rule name can be PASSWORD_LENGTH or PASSWORD_RULE_GROUP, available in HDP Password rules repository.</td>
</tr>
</tbody>
</table>
Sample Server Response

If the default password policy is enabled, the response will take the following format.

```json
{
    "passwordPolicy": {
        "rules": [{
            "ruleId": "pwdLengthRule",
            "ruleName": "PASSWORD_LENGTH_RULE",
            "title": "Contains atleast 8 characters",
            "properties": {
                "minLength": 8,
                "maxLength": 12
            }
        }, {
            "ruleId": "pwdUserNameRule",
            "ruleName": "CHECK_USERNAME_RULE",
            "title": "Password should not contain username",
            "properties": {
                "containsPortionOfUserName": false
            }
        }, {
            "ruleId": "characterRulesGroup",
            "ruleName": "PASSWORD_RULE_GROUP",
            "title": "Can contain characters from these three classes",
            "properties": {
                "minRulesPassed": 3,
                "memberRules": [{
                    "ruleId": "uppercaseLetterRule",
                    "ruleName": "CHARACTER_CLASS_RULE",
                    "title": "Upper Case Letters A-Z",
                    "properties": {
                        "charClass": "[A-Z]",
                        "minChars": 1
                    }
                }, {
                    "ruleId": "lowerCaseLetterRule",
                    "ruleName": "CHARACTER_CLASS_RULE",
                    "title": "Lower Case Letters a-z",
                    "properties": {
                        "charClass": "[a-z]",
                        "minChars": 1
                    }
                }, {
                    "ruleId": "numericRule",
                    "ruleName": "CHARACTER_CLASS_RULE",
                    "title": "Numbers 0-9",
                    "properties": {
                        "charClass": "[0-9]",
                        "minChars": 1
                    }
                }, {
                    "ruleId": "specialCharRule",
                    "ruleName": "CHARACTER_CLASS_RULE",
                    "title": "Non-white space special characters",
                    "properties": {
                        "charClass": "[^A-Za-z0-9]",
                        "nonBlankSpace": true,
                        "minChars": 1
                    }
                }]
            }
        }
    }
}
```
If the Default Password Policy is disabled using System Configuration options, the response will be as follows:

```
Status code: 200
Successful response

{
  "passwordValidationResponse":
    {
      "passed": true
    }
}
```

Sample Server Failure Response

```
{
  "error": {
    "code": 222206007,
    "message": {
      "lang": "en-US",
      "value": "Invalid user ID or password."}
  }
}
```

Authentication/Authorization

No authentication/authorization needed for this API.

Validate Password Policy

Purpose

Validates the password against the password policy.

URL

```
https://<myserver>:<port>/api/public/passwordpolicy/validate
```

Method

POST

URL Parameters

<myserver> is the hostname or IP address of the machine hosting the Hybrid Data Pipeline server for a standalone installation, or the machine hosting the load balancer for a load balancer installation. For a standalone installation, <port> is the port number specified as the Server Access Port during installation. For a load balancer installation, <port> must be either 80 for http or 443 for https. Whenever port 80 or 443 are used, it is not necessary to include the port number in the URL.
Payload Definition

The request takes the following format. The properties of the request are described in the table that follows.

```json
{
   "password": "string"
}
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Valid Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;password&quot;</td>
<td>The new password which needs to be validated against the password policy.</td>
<td><em>Any string value.</em></td>
</tr>
</tbody>
</table>

Sample Request

```json
{
   "password": "TESTUSER"
}
```

Sample Server Response

The response lists details about each rule and whether the password passes or fails each rule.

```json
{
   "passwordValidationResponse": {
      "passed": false,
      "rules": [
         {
            "ruleId": "rule1",
            "title": "Contains at least 8 characters",
            "passed": true
         },
         {
            "ruleId": "rule2",
            "title": "Does not contain portion of username",
            "passed": false
         },
         {
            "ruleId": "rule_char_grp1",
            "title": "Can contain characters from these three classes",
            "passed": false,
            "rules": [
               {
                  "ruleId": "uppercaseLetterRule",
                  "title": "Upper Case Letters A-Z",
                  "passed": true
               },
               {
                  "ruleId": "lowerCaseLetterRule",
                  "title": "Lower Case Letters a-z",
                  "passed": false
               },
               {
                  "ruleId": "numericRule",
                  "title": "Numbers 0-9",
                  "passed": false
               },
               {
                  "ruleId": "special_char_rule",
                  "title": "Non-white space special characters",
                  "passed": false
               }
            ]
         }
      ]
   }
}
```
**Sample Server Response**

```
{
  "error": {
    "code": 222206117,
    "message": {
      "lang": "en-US",
      "value": "Password does not meet the password policy requirements."
    }
  }
}
```

**Authentication/Authorization**

No authentication/authorization needed for this API.

**Hybrid Data Pipeline API Error Messages**

Applications accessing cloud data may encounter error messages, which differ, depending on the data source type you are accessing. Each error message is followed by a possible cause and recommended actions, if applicable.

The following sections describe error messages you may receive back from the Hybrid Data Pipeline Management API. Each error message is followed by a possible cause and recommended actions, if applicable. In addition to general error messages that apply to all components of the Hybrid Data Pipeline Management, additional error messages are returned only by the Data Source or Connector APIs. As with most OData responses, they can either be in XML or JSON, which is determined based on either a `format` parameter or the HTTP header for acceptable media types.

**Table 215: Error message types**

<table>
<thead>
<tr>
<th>Message Source</th>
<th>Error Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common to all APIs</td>
<td>General errors on page 1430</td>
</tr>
<tr>
<td>Data Sources API</td>
<td>Data Sources API error messages on page 1017</td>
</tr>
<tr>
<td>Connector API</td>
<td>Connector API error messages on page 1008</td>
</tr>
<tr>
<td>OAuth API</td>
<td>OAuth API error messages on page 1014</td>
</tr>
<tr>
<td>Administrator API</td>
<td>Administrator API error messages</td>
</tr>
</tbody>
</table>
General errors

The following table describes error messages you may receive back from any of the Hybrid Data Pipeline Management APIs. In most cases, the problem is caused when attempting to send an HTTP request to an invalid URL. See the overview topic for the APIs for the valid requests and URLs:

- Connector API error messages on page 1008
- Data Sources API error messages on page 1017
- OAuth API error messages on page 1014
- Administrator API error messages on page 1430

Table 216: Error Messages for the Hybrid Data Pipeline Management API

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222206007</td>
<td>Invalid user ID or password.</td>
</tr>
<tr>
<td>222206900</td>
<td>Invalid URL for GET: Resource {0} not found.</td>
</tr>
<tr>
<td>222206901</td>
<td>Invalid URL for DELETE: Resource {0} not found.</td>
</tr>
<tr>
<td>222206902</td>
<td>Invalid URL for PUT: Resource {0} not found.</td>
</tr>
<tr>
<td>222206903</td>
<td>Invalid URL for POST: Resource {0} not found.</td>
</tr>
<tr>
<td>222206904</td>
<td>Invalid URL for GET: Resource not specified.</td>
</tr>
<tr>
<td>222206905</td>
<td>Invalid URL for DELETE: Resource not specified.</td>
</tr>
<tr>
<td>222206906</td>
<td>Invalid URL for POST: Resource not specified.</td>
</tr>
<tr>
<td>222206907</td>
<td>Invalid URL for PUT: Resource not specified.</td>
</tr>
</tbody>
</table>
| 222206908    | The method, {0}, is not allowed for this URL, {1}.
| 222206909    | Queries are not supported on this call.          |
| 222208070    | Problem getting version information.              |

Administrator API error messages

This section describes error messages you may receive from the Administrator API. Each error message is followed by a possible cause and recommended actions, if applicable.

Table 217: Error messages for the Administrator API

<table>
<thead>
<tr>
<th>Error code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207901</td>
<td>Problem creating a User at this time. Please try again at another time.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207902</td>
<td>Problem creating a User at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207903</td>
<td>Problem getting Users at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207904</td>
<td>Problem getting a User at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207905</td>
<td>Problem getting a User's StatusInfo at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207906</td>
<td>Problem getting a User's PasswordInfo at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207907</td>
<td>Problem getting a User's Permissions at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207908</td>
<td>Problem updating a User at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207909</td>
<td>Problem updating a User's StatusInfo at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207910</td>
<td>Problem updating a User's PasswordInfo at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222206117</td>
<td>Password does not meet the password policy requirements.</td>
</tr>
<tr>
<td>222207911</td>
<td>Problem updating a User's Permissions at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207912</td>
<td>Problem resetting a User's Password at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207913</td>
<td>Problem changing your password at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207914</td>
<td>Problem getting a User's Details at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207915</td>
<td>Invalid JSON input: <code>{0}</code></td>
</tr>
<tr>
<td><strong>Cause</strong></td>
<td>The specified JSON input was not valid.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td>Correct the JSON input.</td>
</tr>
<tr>
<td>222207916</td>
<td>There is no User with that id: <code>{0}</code>.</td>
</tr>
<tr>
<td><strong>Cause</strong></td>
<td>The UserID specified is not valid.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td>Check the UserID specified in the payload.</td>
</tr>
<tr>
<td>222207917</td>
<td>Problem creating a Role at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207918</td>
<td>Problem deleting a Role at this time. Please try again at another time.</td>
</tr>
<tr>
<td><strong>Cause</strong></td>
<td>The Role could not be deleted.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td>Try deleting the Role later.</td>
</tr>
<tr>
<td>222207919</td>
<td>Problem getting Roles at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207920</td>
<td>Problem getting a Role at this time. Please try again at another time.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207921</td>
<td>Problem updating a Role at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207922</td>
<td>You cannot delete or remove your own account.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: Deleting your own account is not permitted.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Another administrator must remove the account.</td>
</tr>
<tr>
<td>222207923</td>
<td>You cannot change a userName. UserName must remain &quot;{(0)&quot;.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: You tried to change a userName.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Once created, the userName cannot be changed. You can create a new</td>
</tr>
<tr>
<td></td>
<td>User and specify the name that you want to use.</td>
</tr>
<tr>
<td>222207924</td>
<td>There is no Role with that id: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The specified Role does not exist.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Use the Get Roles API to determine the available Roles.</td>
</tr>
<tr>
<td>222208100</td>
<td>&quot;{(0)&quot; value&quot;s length must be between {1} and {2} (inclusive).</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The value's length was not within the specified range.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Correct the value.</td>
</tr>
<tr>
<td>222208101</td>
<td>&quot;{(0)&quot; value&quot;s length must be at least {1}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The value's length did not meet the specified minimum length.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Increase the value's length.</td>
</tr>
<tr>
<td>222208102</td>
<td>&quot;{(0)&quot; value&quot;s length must be no greater than {1}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The value's length was greater than the specified maximum length.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Decrease the value's length.</td>
</tr>
<tr>
<td>222208103</td>
<td>You lack the permissions to access this url.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: You need additional permissions to access the URL.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Ask the Hybrid Data Pipeline administrator to update your permission.</td>
</tr>
<tr>
<td>222208104</td>
<td>LoginId: {0} - lacks the permissions to access this url.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The user with the specified LoginId does not have permissions to access the URL.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Consider increasing the permissions for the user with the specified LoginId.</td>
</tr>
<tr>
<td>222207915</td>
<td>Invalid JSON input: PUT request must contain a body.</td>
</tr>
<tr>
<td>222207916</td>
<td>There is no User with that id: {34}.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207925</td>
<td>Problem processing the limits at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207926</td>
<td>Datasource with id={0} does not belong to user with id={1}.</td>
</tr>
<tr>
<td>222207927</td>
<td>Invalid {0};{1}.</td>
</tr>
<tr>
<td>222207928</td>
<td>Limit not allowed to be set at {0} level.</td>
</tr>
<tr>
<td>222207929</td>
<td>Limit value not in range (({0},{1}).</td>
</tr>
<tr>
<td>222207930</td>
<td>Limit does not exist for id = {1}.</td>
</tr>
<tr>
<td>222207931</td>
<td>Limit already exists for id = {1}.</td>
</tr>
<tr>
<td>222207936</td>
<td>Invalid Driver Logger name: {abc}. Allowed Values are adapter, sql, drivercommunication, cloud (case insensitive).</td>
</tr>
<tr>
<td>222207932</td>
<td>There is no DataSource with id : {39}</td>
</tr>
<tr>
<td>222207938</td>
<td>Invalid DAS Log Level: {abc}. Allowed Values are SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST (case insensitive).</td>
</tr>
<tr>
<td>222207939</td>
<td>Invalid Driver Log Level: {abc} for Logger adapter. Allowed Values are SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST (case insensitive).</td>
</tr>
<tr>
<td>222208029</td>
<td>Duplicate Entry found for Logger : {abc}</td>
</tr>
</tbody>
</table>

**Connector API error messages**

The following section describes error messages you may receive back from the Hybrid Data Pipeline Connector API. Each error message is followed by a possible cause and recommended actions, if applicable.

**Table 218: Error messages for the Connector API**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222206850</td>
<td>The label {0} is already used by other connector. Please use another label. <strong>Cause:</strong> The specified label has already been defined by another Connector. The label must be unique. <strong>Action:</strong> Modify the label so that it is unique.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207100</td>
<td>Problem getting the users from the Access Control List at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207101</td>
<td>Problem adding the user(s) to the Access Control List at this time. Please try again at another time.</td>
</tr>
</tbody>
</table>
| 222207102   | Invalid user name: {0}.  
  **Cause:** The user name in the request payload is not valid.  
  **Action:** Make sure the user name in the request payload has the appropriate permissions and is specified correctly. |
| 222207103   | There is a problem with the JSON input: Owners -- {0}, {1}--do not match  
  **Cause:** The JSON statement is not correct.  
  **Action:** Check the Owners in the JSON input. |
| 222207104   | Problem getting the Connector from the Access Control List at this time. Please try again at another time. |
| 222207106   | The number of users specified ({0}) exceeds the system limit ({1}). Please use multiple requests.  
  **Cause:** Only one user can be specified.  
  **Action:** Create a separate request for each user. |
| 222207107   | Invalid JSON input: {0}  
  **Cause:** The specified JSON input was not valid.  
  **Action:** Correct the JSON input. |
| 222207108   | 'authUser' was not supplied or was not an array.  
  **Cause:** The request must specify an authUser parameter.  
  **Action:** Add an authUser array. The array can be empty. |
| 222207109   | Problem getting the connector info for {0}. Please try again at another time.  
  **Cause:** A problem occurred when getting the Connector information for the specified Connector.  
  **Action:** Please try again at another time. |
| 222207110   | Problem updating users for {0}. Please try again at another time.  
  **Cause:** A problem occurred when updating users for the specified Connector.  
  **Action:** Please try again at another time. |
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 222207111  | Problem deleting the user(s) from the Access Control List at this time. Please try again at another time.  
**Cause:** A problem occurred when deleting users from the specified Connector.  
**Action:** Please try again at another time. |
| 222207112  | Connector {0} does not exist or you are not the owner.  
**Cause:** Either the specified On-Premises Connector does not exist, or you are not the owner of the Connector.  
**Action:** The owner specified in the request must match the current owner of the Connector or Connector Group. Changing the owner of a Connector or Connector Group is not supported. |
| 222207115  | Problem getting the Connector info. Please try again at another time.  
**Cause:** A problem occurred when getting the Connector information.  
**Action:** Try the operation later. |
| 222207116  | Problem deleting the Connector.  
**Cause:** A problem occurred deleting the Connector.  
**Action:** Try the operation later. |
| 222207117  | 'members' was not supplied.  
**Cause:** The Connector is a GroupConnector, and must contain a connectorGroup object that contains a members array.  
**Action:** A Connector Group must contain a connectorGroup object that contains a members array. The members array was not defined in the connectorGroup. |
| 222207118  | 'memberID' was not supplied.  
**Cause:** The members array for this GroupConnector must contain a member_id parameter.  
**Action:** Check the connectorGroup object. The members array must contain a memberID. |
| 222207119  | 'sequence' was not supplied.  
**Cause:** The members array for this GroupConnector must contain a sequence parameter.  
**Action:** Check the connectorGroup object. The members array must contain a sequence. |
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
<th>Cause</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207121</td>
<td>You cannot delete the last member of the Connector Group(s): {0}.</td>
<td>The JSON statement attempted to remove the last member of a Connector Group.</td>
<td>You cannot delete the last member of the Connector Group. To delete a Connector Group, use the Delete Group API.</td>
</tr>
<tr>
<td>222207122</td>
<td>Problem deleting members. Please try again at another time.</td>
<td>A problem occurred when deleting members from a group.</td>
<td>Try the operation later.</td>
</tr>
<tr>
<td>222207123</td>
<td>Problem getting members. Please try again at another time.</td>
<td>A problem occurred when getting members.</td>
<td>Try the operation later.</td>
</tr>
<tr>
<td>222207124</td>
<td>ConnectionTimeout must have a value with a minimum of 1.</td>
<td>ConnectionTimeout wasn't set to a positive integer.</td>
<td>Set ConnectionTimeout to a positive integer, 1 or greater.</td>
</tr>
<tr>
<td>222207125</td>
<td>RetryDelay must have a value with a minimum of 0.</td>
<td>RetryDelay was set to an invalid value.</td>
<td>Set RetryDelay to 0 or a positive integer. See &quot;Update Connector Information&quot; for more information.</td>
</tr>
<tr>
<td>222207126</td>
<td>There must be at least one member in a Group Connector at all times.</td>
<td>You attempted to delete the last member of a Group Connector.</td>
<td>A Group Connector must contain at least one member.</td>
</tr>
<tr>
<td>222207127</td>
<td>Problem creating a ConnectorId. Please try again at another time.</td>
<td>A problem occurred when creating a Connector ID.</td>
<td>Try the operation later.</td>
</tr>
<tr>
<td>222207128</td>
<td>This is not a valid payload for an update. Please consult the documentation.</td>
<td>The JSON statement was not valid for an update.</td>
<td>Check the JSON statement.</td>
</tr>
<tr>
<td>222207129</td>
<td>You cannot change the ConnectorId.</td>
<td>You cannot change the ConnectorID.</td>
<td>The Connector ID is generated by Hybrid Data Pipeline and is specific to each Connector. It cannot be changed.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207130</td>
<td>You cannot change the owner of the Connector.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> You cannot change the owner of the Connector. Only the owner of the Connector can reassign the Connector to a different owner.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Consult the Hybrid Data Pipeline administrator.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207131</td>
<td>You cannot add a Group Connector, <code>{0}</code> to another Group.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The specified Connector has been defined as a Group Connector. You cannot add a Group Connector to another Group.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Use the Connector ID for a Connector that is not a Group Connector.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207132</td>
<td>This Connector <code>{0}</code> is not a member of Connector <code>{1}</code>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The request specified a Connector that is not a member of the Group Connector.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Change the request to use a member of the Group Connector.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207133</td>
<td>Problem adding connector(s) to the group connector at this time. Please try again at another time.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when adding one or more Connectors to the group connector.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try the operation later.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207134</td>
<td>Problem updating connector(s) to the group connector at this time. Please try again at another time.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when creating a Connector ID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try the operation later.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207135</td>
<td>Problem determining authorization to use connector. Please try again at another time.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when determining authorization to use the Connector.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try the operation later.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207136</td>
<td>Problem getting connector statistics at this time. Please try again at another time.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when getting connector statistics.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try the operation later.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222207137</td>
<td><code>{0}</code> is not a supported Load Balancing type. Please refer to the documentation on Load Balancing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The JSON input specified an invalid type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> See &quot;Enable Round-Robin load balancing for a group&quot; for valid types for load balancing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 222207138  | ConnectorId {0} is already in the following GroupConnector: {1}. A ConnectorId can only be in one GroupConnector.  
**Cause:** The specified Connector is already a member of a Connector group.  
**Action:** Add a different Connector to the Connector group.                                                                                                                                                        |
| 222207139  | Problem updating a ConnectorId. Please try again at another time.  
**Cause:** A problem occurred when updating the Connector information.  
**Action:** Try the operation later.                                                                                                                                                                               |
| 222207140  | ConnectorId {0} is already in the current GroupConnector {1}. If POST - this GroupConnector failed to be created.  
**Cause:** The specified Connector is already a member of the current Connector group. If you submitted a POST request, the operation failed. The GroupConnector was not created.  
**Action:** Add a different On-Premises Connector to the Connector group.                                                                                         |
| 222207141  | Problem getting version and owner. Please try again at another time.  
**Cause:** A problem occurred when getting version and owner.  
**Action:** Try the operation later.                                                                                                                                                                               |
| 222207142  | Problem getting authorized users. Please try again at another time.  
**Cause:** A problem occurred when getting authorized users.  
**Action:** Try the operation later.                                                                                                                                                                               |
| 222207143  | Sequence must be an INTEGER greater than 0.  
**Cause:** The sequence parameter must be greater an integer greater than 0.  
**Action:** Check the value of the sequence parameter.                                                                                                                                                         |
| 222207144  | Weight must be an INTEGER greater than 0.  
**Cause:** The weight parameter must be greater an integer greater than 0.  
**Action:** Check the value of the weight parameter.                                                                                                                                                         |
| 222207145  | Problem getting modified connectors.                                                                                                                                                                                                                                                                                                      |
| 222207146  | Connector (0) (Version {1}) does not support Load Balancing. Only version 3.0 and higher support Load Balancing. Please update to the latest version.  
**Cause:** The specified Connector is Version 1.0, and doesn’t support Load Balancing.  
**Action:** Update the Connector to Version 3.0 or higher. See the Progress DataDirect Hybrid Data Pipeline Installation Guide for more information. |
**Data Sources API error messages**

This section describes error messages you may receive from the Data Sources API. Each error message is followed by a possible cause and recommended actions, if applicable.

**Table 219: Error messages for the Data Sources API**

<table>
<thead>
<tr>
<th>Error code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207000</td>
<td>Problem updating your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207001</td>
<td>Problem retrieving your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207002</td>
<td>Invalid DataSource Option: {0}.</td>
</tr>
<tr>
<td>222207003</td>
<td>There is a problem connecting to the DataSource. {0}</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207004</td>
<td>There is no DataSource with that id: {0}. <strong>Cause</strong>: The DataSource ID is incorrect. The data source ID may have been entered incorrectly, or the data source ID might have been invalidated by the administrator. <strong>Action</strong>: Correct the DataSource ID.</td>
</tr>
<tr>
<td>222207005</td>
<td>Expected values for connectType : 'Cloud' / 'Hybrid'. Your value was {0}. Please try again with proper value. <strong>Cause</strong>: The connectionType parameter specified a value other than Cloud or Hybrid. <strong>Action</strong>: Specify a valid value.</td>
</tr>
<tr>
<td>222207006</td>
<td>Problem deleting your DataSource at this time. Please try again at another time. <strong>Cause</strong>: The DataSource couldn't be deleted at this time. <strong>Action</strong>: Try again later.</td>
</tr>
<tr>
<td>222207007</td>
<td>Invalid JSON Input: {0} <strong>Cause</strong>: The JSON input was not valid. <strong>Action</strong>: Correct the JSON statement and retry the query.</td>
</tr>
<tr>
<td>222207008</td>
<td>connectionType is not allowed to be changed. It must remain : {0}.</td>
</tr>
<tr>
<td>222207009</td>
<td>Expected values for map:'refresh'/recreate'/none'. Your value was {0}. Please try again with proper value. <strong>Cause</strong>: The map parameter specified an invalid value. <strong>Action</strong>: Change the value for the map parameter. The valid values are refresh, recreate, and none.</td>
</tr>
<tr>
<td>222207010</td>
<td>Missing 'connectionType' in payload. <strong>Cause</strong>: The connectionType parameter is missing, or no value was defined. <strong>Action</strong>: Check the payload. Add the connectionType and a valid value.</td>
</tr>
<tr>
<td>222207011</td>
<td>Invalid DataSource ID: {0}. <strong>Cause</strong>: The specified DataSource ID is invalid. <strong>Action</strong>: Check the DataSource ID.</td>
</tr>
<tr>
<td>222207012</td>
<td>You are not authorized to create a DataSource with this DataStore id: {0}. Please contact Technical Support if you would like to upgrade your account. <strong>Cause</strong>: The DataStore you specified is not included in your subscription plan, or you are not authorized to use the DataStore. For example, the Hybrid Data Pipeline administrator might have limited the number of users who can access Salesforce. <strong>Action</strong>: Contact your Hybrid Data Pipeline administrator or Technical Support.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207013</td>
<td>Problem validating your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> There was a problem validating your DataSource.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try validating your DataSource later.</td>
</tr>
<tr>
<td>222207014</td>
<td>You already have a DataSource with the name {0}. Please choose another name.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A data source with that name already exists.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Choose a different name for the data source.</td>
</tr>
<tr>
<td>222207015</td>
<td>Invalid DataStore ID: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The DataStoreID specified is not valid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the DataStoreID specified in the payload. You can get the</td>
</tr>
<tr>
<td></td>
<td>DataSourceID from the DataStores resource.</td>
</tr>
<tr>
<td>222207016</td>
<td>Missing 'name' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The name parameter is not in the payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> The name parameter is required. The name must contain only alphabetic</td>
</tr>
<tr>
<td></td>
<td>characters and the underscore, and must begin with a letter.</td>
</tr>
<tr>
<td>222207017</td>
<td>Problem refreshing your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The DataSource could not be refreshed.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try refreshing the DataSource later.</td>
</tr>
<tr>
<td>222207018</td>
<td>{0} is an unrecognized argument for /map. Expected 'map' and/or 'model' only.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> An unrecognized argument was used for map.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> The only valid arguments are map and model.</td>
</tr>
<tr>
<td>222207019</td>
<td>Missing 'id' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The id property is the data source id used to reference the data source in</td>
</tr>
<tr>
<td></td>
<td>the Hybrid Data Pipeline Management API URLs.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add the data source id for the data source.</td>
</tr>
<tr>
<td>222207020</td>
<td>Missing 'password' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The password property is missing.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the payload and add a valid password.</td>
</tr>
<tr>
<td>222207021</td>
<td>DataStore is not allowed to be changed. It must remain: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The DataStore value cannot be changed.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the JSON string.</td>
</tr>
<tr>
<td>222207022</td>
<td>There was a problem deleting the DataSource. Multiple rows were somehow deleted.</td>
</tr>
<tr>
<td></td>
<td>{0}</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207023</td>
<td>Problem connecting to your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem connecting to the data source.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try connecting later.</td>
</tr>
<tr>
<td>222207024</td>
<td>Problem retrieving your DataSources at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem retrieving your data sources.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207025</td>
<td>Problem creating your DataSource at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem creating your data sources.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207026</td>
<td>Missing 'dataStore' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The payload did not specify a valid dataStore element.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Add the <code>dataStore</code> to the payload.</td>
</tr>
<tr>
<td>222207027</td>
<td>There is a problem getting the DataStore(s) at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: There was a problem getting your data sources.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try the operation later.</td>
</tr>
<tr>
<td>222207028</td>
<td>Missing 'userId' in payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The <code>user</code> parameter was not in the payload, or no value was defined.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Make sure the payload contains the <code>user</code> parameter with a valid user name.</td>
</tr>
<tr>
<td>222207029</td>
<td>Expected values for model: 'refresh' / 'none'. Your value was (0). Please try again with proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The <code>model</code> parameter specified an invalid parameter.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the <code>model</code> parameter and change the value. The valid values are <code>refresh</code> and <code>none</code>.</td>
</tr>
<tr>
<td>222207030</td>
<td>Data Source 'id' in the JSON Request must match the resource. i.e. /datasources/&lt;id&gt;. DataSource 'id' is an optional field.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The data source ID is generated by Hybrid Data Pipeline and cannot be changed.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Including the data source ID in the JSON request is optional. When the ID is included in the JSON request, make sure it matches the resource.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207031</td>
<td>Invalid userName {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The specified user name is not valid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Enter a valid user name.</td>
</tr>
<tr>
<td>222207032</td>
<td>Must supply 'map' and/or 'model' in your payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: Either map or model must be specified in the payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Add map or model to the payload.</td>
</tr>
<tr>
<td>222207033</td>
<td>Problem retrieving the members of your DataSource Group. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try again later.</td>
</tr>
<tr>
<td>222207034</td>
<td>Problem updating the members of your DataSource Group. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try again later.</td>
</tr>
<tr>
<td>222207035</td>
<td>Problem creating one or more new member DataSources for your DataSource Group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try again later.</td>
</tr>
<tr>
<td>222207036</td>
<td>Problem removing one or more member DataSource from your DataSource Group. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: A problem occurred when attempting to remove one or more member data sources from your data source group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try removing the member data sources from the data source group later.</td>
</tr>
<tr>
<td>222207037</td>
<td>Only DataSource Groups can have member DataSources assigned.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: An attempt was made to add a member data source to a data source that was not defined as a data source group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Add the member DataSource to a data source group.</td>
</tr>
<tr>
<td>222207038</td>
<td>DataSource {0} must be a DataSource Group when used in this way.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: An attempt was made to use a simple or member data source as a data source group.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: You can't change the data source into being a data source group. Specify a data source that is a data source group for this action.</td>
</tr>
<tr>
<td>222207039</td>
<td>DataSource {0} cannot be a DataSource Group when used in this way.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: An attempt was made to use a data source group when a simple or member DataSource was needed.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Use a simple data source or a member data source.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207040</td>
<td>An existing DataSource {0} was seen while adding new DataSource members to a DataSource Group.</td>
</tr>
</tbody>
</table>
| 222207041   | The DataSource cannot be removed because it is used in one or more DataSource Groups: {0}.  
**Cause:** An attempt was made to delete a data source that is a member of one or more data source group.  
**Action:** Remove the data source from each data source group that it is a member of. |
| 222207042   | When updating a DataSource Group, a "members" section must be supplied.  
**Cause:** An attempt was made to update a data source group, but the payload did not contain a members parameter.  
**Action:** Add a members parameter to the options object. |
| 222207043   | You are not authorized to update a {0} DataSource (DataStore id: {1}). Please contact Customer Support if you would like to upgrade your account.  
**Cause:** You are not authorized to update the specified data source for the data source type.  
**Action:** Check with your Hybrid Data Pipeline administrator to see if the authorization can be changed. For example, the subscription might be configured for 5 users to update Salesforce. |
| 222207044   | A DataSource Group connectionType must be 'Group'. Your value was {0}. Please try again with the proper value.  
**Cause:** The value specified for connectionType was invalid for a data source group.  
**Action:** Change the value of connectionType to Group. |
| 222207045   | MaximumEntityNameLength must be an integer between 10 and 128 inclusive, but your value was {0}. Please try again with the proper value.  
**Cause:** The value specified for MaximumEntityNameLength was not an integer between 10 and 128 inclusive.  
**Action:** Specify an integer between 10 and 128 inclusive. |
| 222207046   | MaximumEntityNameLength is outside the valid range of 10 to 128 inclusive. but your value was {0}. Please try again with the proper value.  
**Cause:** The value specified for MaximumEntityNameLength was not in the valid range.  
**Action:** Specify an integer between 10 and 128, inclusive. |
<table>
<thead>
<tr>
<th>Error code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207047</td>
<td>The entity prefix for member datasources must be specified. For source {0}, it was not. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> Each member data source must specify a unique entity prefix.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify a unique entity prefix that is less than half the length of the value specified for MaximumEntityNameLength.</td>
</tr>
<tr>
<td>222207048</td>
<td>The entity prefix for source {0} must be less than half the maximum entity name length. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The entity prefix for the specified data source must specify a unique entity prefix that is less than half the maximum entity name length.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify a unique entity prefix that is less than half the length of the value specified for MaximumEntityNameLength.</td>
</tr>
<tr>
<td>222207049</td>
<td>All of the entity prefixes within a DataSource Group must be unique. DataSource {0} has a duplicate. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> Each member data source must specify a unique entity prefix.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Specify a unique entity prefix that is less than half the length of the value specified for MaximumEntityNameLength.</td>
</tr>
<tr>
<td>222207050</td>
<td>Entity prefixes cannot contain underscores, but DataSource {0} has one. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The entity prefix can contain only alphanumeric characters and can't contain an underscore.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Modify the entity prefix.</td>
</tr>
<tr>
<td>222207051</td>
<td>The entity prefix name for member DataSource {0} does not follow OData guidelines. Please try again with the proper value.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The entity prefix can contain only alphanumeric characters and must begin with an alphabetic character.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Correct the entity prefix.</td>
</tr>
<tr>
<td>222207052</td>
<td>Problem getting the status of your OData Model Creation. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when getting the status of the OData Model Creation.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try again later.</td>
</tr>
<tr>
<td>222207053</td>
<td>Problem starting creation of your OData Model. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> A problem occurred when starting to create your OData model.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try again later.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207054</td>
<td>Cannot start the OData Model Creation because it is currently running. Please see the documentation if you wish to restart the creation.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The OData Model creation operation is already running.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Try again later.</td>
</tr>
<tr>
<td>222207055</td>
<td>The status was changed during the process of the request. Please verify and send request again if needed.</td>
</tr>
<tr>
<td>222207056</td>
<td>You cannot create an OData Model for a DataSource Group.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: An attempt was made to create an OData model for a DataSource Group. You can only create an OData model for a simple data source.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the members of the data source group and make sure that each has an OData model.</td>
</tr>
<tr>
<td>222207057</td>
<td>You cannot refresh/recreate the map of a DataSource Group.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: An attempt was made to refresh or create the map for a DataSource Group. You can only refresh or create a map for a simple data source.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Refresh or create the map for the member data sources in the DataSource Group.</td>
</tr>
<tr>
<td>222207058</td>
<td>DataSource {0} must have an OData map.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: A schema map has not been defined for the data source.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: The data source must be enabled for OData by defining a schema map.</td>
</tr>
<tr>
<td>222207059</td>
<td>Test connect cannot be performed on a DataSource Group. To test connectivity, the member data sources of the group should be tested.</td>
</tr>
<tr>
<td>222207060</td>
<td>There are duplicate members in the payload. Please remove the duplicates and try again.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The payload contains duplicate members.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Remove the duplicate members and try again</td>
</tr>
<tr>
<td>222207061</td>
<td>Member {0} already exists in the DataSource Group that matches one in your payload; please adjust your payload and try again.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The specified member already exists in the DataSource Group specified in the payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the payload, and remove or replace the duplicate member.</td>
</tr>
<tr>
<td>222207062</td>
<td>The schema {0} does not exist.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause</strong>: The specified schema does not exist.</td>
</tr>
<tr>
<td></td>
<td><strong>Action</strong>: Check the schema name. If necessary, use the Get Schemas API for a list of valid schemas.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| 222207063  | The table \{0\} does not exist under schema \{1\}.  
**Cause:** The specified table does not exist under the specified schema.  
**Action:** Check the table name and schema name. |
| 222207064  | Problem retrieving the schemas at this time. Please try again at another time. |
| 222207065  | Problem retrieving the tables at this time. Please try again at another time. |
| 222207066  | Problem retrieving the columns at this time. Please try again at another time. |
| 222207067  | Problem retrieving the primary keys at this time. Please try again at another time. |
| 222207068  | Problem retrieving the table details at this time. Please try again at another time. |
| 222207069  | Invalid OAuthProfileId: \{0\}.  
**Cause:** The specified OAuthProfileID is not valid.  
**Action:** Correct the OAuthProfileID. |
| 222207070  | The OAuthProfile data store \{(0)\} does not match the DataSource data store\{(1)\}  
**Cause:** The specified OAuthProfile data source type does not match the data source type specified in the DataSource.  
**Action:** Check the OAuthProfile data source type and the DataSource data source type. |

**HTTP Response Codes Returned by the Hybrid Data Pipeline Management Data Sources API**

Hybrid Data Pipeline Management Data Sources API returns standard HTTP response codes as described in the following table, under the conditions listed in the description. The descriptions differ somewhat from the general description found earlier in this document.

**Table 220: HTTP Error Messages for the Data Sources API**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 200        | OK          
  The request was successfully completed. If this request created a new resource that is addressable with a URI, and a response body is returned containing a representation of the new resource, a 200 status will be returned with a location header containing the canonical URI for the newly created resource. |
| 201        | Created     
  A request that created a new resource was completed and no response body containing a representation of the new resource is being returned. A location header containing the canonical URI for the newly created resource will be returned. |
### OAuth API Error Messages

This section describes error messages you may receive from the OAuth API. Each error message is followed by a possible cause and recommended actions, if applicable.

#### Table 221: Error Messages for the OAuth API

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>222207700</td>
<td>Problem creating an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207701</td>
<td>Problem deleting an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207702</td>
<td>Problem getting OAuthProfiles at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207703</td>
<td>Problem getting an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207704</td>
<td>Problem updating an OAuthProfile at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207705</td>
<td>Problem creating an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207706</td>
<td>Problem deleting an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The OAuthApplication couldn't be deleted at this time.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Try again later.</td>
</tr>
<tr>
<td>222207707</td>
<td>Problem getting OAuthApplications at this time. Please try again at another time.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207708</td>
<td>Problem getting an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207709</td>
<td>Problem updating an OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207710</td>
<td>Invalid OAuthProfileId: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The OAuthProfileId parameter is missing, or no value was defined.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the payload. Add the OAuthProfileId parameter with a valid value.</td>
</tr>
<tr>
<td>222207711</td>
<td>Invalid OAuthApplicationId: {0}.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The specified OAuthApplicationId is invalid.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the OAuthApplicationId.</td>
</tr>
<tr>
<td>222207712</td>
<td>Missing 'name' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The name parameter for the OAuthApplication is required, but none was specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add a value for name, that is, the name of the OAuthApplication. The name can contain only alphanumeric characters and the underscore character.</td>
</tr>
<tr>
<td>222207713</td>
<td>Missing 'dataStore' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The dataStore parameter is required, but none was specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add a value for dataStore, that is, the name of the dataStore. The dataStore ID can be obtained from the &lt;base&gt;/datastores resource.</td>
</tr>
<tr>
<td>222207714</td>
<td>Missing 'oauthAppId' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The oauthAppId parameter is required, but none was specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Add a value for oauthAppId. This property is generated by Hybrid Data Pipeline and cannot be changed once assigned. The ID is used to identify the data source type in data source references.</td>
</tr>
<tr>
<td>222207715</td>
<td>Missing 'refreshToken' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The refreshToken was not specified.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> Check the refreshToken specified in in the payload.</td>
</tr>
<tr>
<td>222207716</td>
<td>Missing 'clientId' from payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Cause:</strong> The clientId parameter is not in the payload.</td>
</tr>
<tr>
<td></td>
<td><strong>Action:</strong> The clientId parameter is required. Visit the Google Developers Console to obtain OAuth 2.0 credentials that are known to both Google and your application.</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>222207717</td>
<td>Missing 'clientSecret' from payload.</td>
</tr>
<tr>
<td>222207718</td>
<td>Problem validating the OAuthApplication at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207719</td>
<td>OAuthProfile name must be unique for a given OAuthApplication.</td>
</tr>
<tr>
<td>222207720</td>
<td>That OAuthApplication Name is invalid. Please choose another name.</td>
</tr>
<tr>
<td>222207721</td>
<td>You cannot change the DataStore of a OAuthApplication.</td>
</tr>
<tr>
<td>222207722</td>
<td>Problem getting the OAuthProfile Statistics at this time. Please try again at another time.</td>
</tr>
<tr>
<td>222207723</td>
<td>DataStore {0} does not support OAuth.</td>
</tr>
</tbody>
</table>

**HTTP Response Codes Returned by the Hybrid Data Pipeline Management Data Sources API**

Hybrid Data Pipeline Management Data Sources API returns standard HTTP response codes as described in the following table, under the conditions listed in the description. The descriptions differ somewhat from the general description found earlier in this document.
### Table 222: HTTP Error Messages for the Data Sources API

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
</tr>
<tr>
<td></td>
<td>The request was successfully completed. If this request created a new resource that is addressable with a URI, and a response body is returned containing a representation of the new resource, a 200 status will be returned with a location header containing the canonical URI for the newly created resource.</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
</tr>
<tr>
<td></td>
<td>A request that created a new resource was completed and no response body containing a representation of the new resource is being returned. A location header containing the canonical URI for the newly created resource will be returned.</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
</tr>
<tr>
<td></td>
<td>The JSON request is invalid.</td>
</tr>
<tr>
<td>401</td>
<td>Not Authorized</td>
</tr>
<tr>
<td></td>
<td>The user is not authorized. An invalid user name and/or password was used.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
</tr>
<tr>
<td></td>
<td>The &lt;DataSource&gt; was not found, where &lt;resource_type&gt; is DataSource.</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td></td>
<td>The server encountered an unexpected condition which prevented it from fulfilling the request.</td>
</tr>
<tr>
<td>501</td>
<td>Not Implemented</td>
</tr>
<tr>
<td></td>
<td>The server currently does not support the functionality required to fulfill the request.</td>
</tr>
</tbody>
</table>