Progress®



Use Pro2

OpenEdge®

Table of Contents

Part I: Get started....9

What's new in Pro2....9
Get started with Pro2....11
Install and configure Pro2 for LAN....15
Install and configure Pro2 for WAN....29
Install and configure Pro2 on Linux....44
Pro2 Enterprise View....48

Part II: Manage....53

Customize Pro2....54
Upgrade Pro2 5.5 to 6.2....55
Uninstall Pro2....60
Use replication triggers....61
Remove replication triggers....64
Customize Pro2 replication....64
Split replication threads....65
Configure Pro2 to target SQL database connectivity....69
Configure Pro2 e-mail notifications....73
Use second-pass replication....74
Bulk load with Pro2....76
Dump and load with Pro2....83
Migrate your OpenEdge database with Pro2....83
Check replication logs....86
How to schedule jobs in Pro2....87
Use CDC with Pro2....90
Manage Pro2 Properties....98
GET bulk load report API....100
GET replication queue records API....105
GET thread data API....107
GET Pro2 connection information API....109
GET version information API....109
GET instance information API....110

Part III: Additional information....113

Pro2 FAQs....113
Command-line references....114
Processor procedure generator....118
Pro2 repl databases and table schemas....118

Pro2 directory hierarchy....124
Pro2 program files....126
Replication procedure library....129
Replication processor....130
Plan for disaster recovery with Pro2....131

Copyright

© 2021 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.

DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Defrag This, Deliver More Than Expected, DevReach (and design), Icenium, Inspec, Ipswitch, iMacros, Kendo UI, Kinvey, MessageWay, MOVEit, NativeChat, NativeScript, OpenEdge, Powered by Chef, Powered by Progress, Progress, Progress Software Developers Network, SequeLink, Sitefinity (and Design), Sitefinity, Sitefinity (and design), SpeedScript, Stylus Studio, Stylized Design (Arrow/3D Box logo), Styleized Design (C Chef logo), Stylized Design of Samurai, TeamPulse, Telerik, Telerik (and design), Test Studio, WebSpeed, WhatsConfigured, WhatsConnected, WhatsUp, and WS_FTP 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, Chef Automate, Chef Compliance, Chef Desktop, Chef Habitat, Chef WorkStation, Corticon.js, Corticon Rules, Data Access, DataDirect Autonomous REST Connector, DataDirect Spy, DevCraft, Fiddler, Fiddler Everywhere, FiddlerCap, FiddlerCore, FiddlerScript, Hybrid Data Pipeline, iMail, JustAssembly, JustDecompile, JustMock, KendoReact, NativeScript Sidekick, OpenAccess, PASOE, Pro2, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, Push Jobs, SafeSpaceVR, Sitefinity Cloud, Sitefinity CMS, Sitefinity Digital Experience Cloud, Sitefinity Feather, Sitefinity Insight, Sitefinity Thunder, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Supermarket, SupportLink, Unite UX, 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.

July 2021

Product version: Progress OpenEdge 12.4

Preface

Purpose

This manual explains how to use OpenEdge Pro2 to replicate your OpenEdge database to another database of your choosing. It provides installation and configuration instructions for Pro2 utilities including the replication process, replication triggers, Change Data Capture, bulk loading, and more. The manual is composed of three parts: Get started, Manage, and Additional information. Each part is dedicated to its respective title. The Get started section describes installation and initial configuration tasks as well as deployment architecture. The Manage section contains topics that address post installation tasks such as customizing Pro2 or upgrading to a new release. Lastly, the Additional information section describes reference information like FAQs, command-line references and schema information.

For the latest documentation updates see the OpenEdge Product Documentation at https://docs.progress.com.

Audience

This document is intended for database administrators and consultants to help them configure Pro2 and use it to replicate data. It assumes a fundamental knowledge of both OpenEdge and DataServer for Microsoft SQL and Oracle.

Organization

Get started with Pro2

OpenEdge Pro2 is a lightweight, configurable tool that provides near real-time replication of an OpenEdge database to Microsoft SQL, Oracle, or another OpenEdge database.

Install and configure Pro2 for LAN on page 15

Illustrates how to configure and deploy Pro2 for a local area network.

Install and configure Pro2 for WAN on page 29

Describes how to configure and deploy Pro2 for a wide area network.

Install and configure Pro2 on Linux on page 44

how to configure and deploy Pro2 for a wide area network.

Pro2 Enterprise View on page 48

Describes Pro2 Enterprise View and how to install and configure it.

Customize Pro2 on page 54

Elaborates on how to customize Pro2 to suit your business needs.

Upgrade Pro2 6.0 to 6.2

Describes how to upgrade Pro2.

Uninstall Pro2 on page 60

Walks through the steps of uninstalling Pro2.

Use replication triggers on page 61

Describes what replication triggers are, and how to use them.

Remove replication triggers on page 64

Illustrates how to removes replication triggers.

Customize Pro2 replication on page 64

Describes how you can customize Pro2.

Automate replications

Discusses replication and how it can be automated.

Split replication threads on page 65

Describes how to split replication threads.

Configure OpenEdge to SQL connectivity for Pro2

Discusses SQL databases and you can configure them for Pro2

Configure Pro2 e-mail notifications on page 73

Describes how to set up, reset, and remove Pro2 replication failure email notifications.

Use second-pass replication on page 74

Illustrates how to use second pass replication with Pro2

Bulk load with Pro2 on page 76

Overviews bulk-loading with Pro2

Check replication logs on page 86

Explains how use the check replication log function.

How to schedule jobs in Pro2 on page 87

Describes how to use the Job Scheduler in the Pro2 UI.

Use CDC with Pro2 on page 90

Discusses Change Data Capture and how to use it with Pro2.

Documentation conventions

See Documentation Conventions for an explanation of the terminology, format, and typographical conventions used throughout the OpenEdge content library, including information about the following:

- Using ABL documentation
- Examples of syntax descriptions
- OpenEdge messages

Get started

For details, see the following topics:

- What's new in Pro2
- Get started with Pro2
- Install and configure Pro2 for LAN
- Install and configure Pro2 for WAN
- Install and configure Pro2 on Linux
- Pro2 Enterprise View

What's new in Pro2

Pro2 6.2 release

The Pro2 6.2 release offers new features and enhancements that increase usability and management.

Pro2 6.2 database migration

To stay competitive in the modern business world, your applications need to run as close to 24/7 as possible. In some cases, you can only have downtime limited to 4 hours. This can make upgrading or migrating your OpenEdge database difficult, even with proper planning. With Pro2 version 6.2, you can migrate your OpenEdge database with minimal downtime. In this video, we will show you how you can migrate with Pro2.

Pro2 6.2 CDC enhancements

In previous versions of Pro2, the CDC replication process converted CDC records to Pro2 `replqueue` records. This procedure increased the total time of the replication.

In Pro2 6.2, CDC replication records are consumed directly by Pro2, bypassing the need for the records to be converted to `replqueue` records.

The direct replication of CDC works on all Pro2 target database types: Microsoft SQL Server, Oracle, and OpenEdge. This enhancement also functions on both network topologies LAN and WAN.

For more information, see Use CDC with Pro2 on page 90.

Pro2 6.2 Bulk-load enhancements

In previous releases of Pro2 bulk load process used row by row transactional commits on target replication database.

In Pro2 6.2, the bulk-load process is optimized to directly generate native SQL Insert statements and wrap multiple inserts into a single target transaction. This entrancement has improved the efficiency of the load process. For table loads that run for days or weeks, this overall improvement can be dramatic.

Because the new bulk-load method is a Pro2 produced methodology, this new feature operates identically in both Oracle and Microsoft SQL Server environments, allowing Pro2 to provide wide-transaction support in Microsoft SQL Server environments.

For more information, see Bulk load with Pro2 on page 76

Pro2 6.1 release

The Pro2 6.1 release offers new features and enhancements that increase usability and management.

Database schema changes in Pro2 6.1

The repl.db schema remains unchanged from 6.0. However, the 5.5 and 6.0 dashboard database, ReplPerf.db, has been deprecated in favor of a new database labeled pro2. This database resides only within the Pro2 installation folder and now holds configuration information as well as newer performance metrics.

Installation updates in Pro2 6.1

The Pro2 web application is now deployed using .paar files in place of .war files. The newer method eliminates installation issues encountered with different OpenEdge versions.

The Enterprise View in Pro2 6.1

The Pro2 Enterprise View is a centralized dashboard that aggregates mission-critical data, reports, and metrics across all your Pro2 instances. From the Enterprise View dashboard, you can quickly access each of your Pro2 instances to manage and act on any reports or replications from the instance. The Enterprise dashboard utilizes a new database labeled Pro2Ent and does not require connectivity to any source or target databases. For more information, see Pro2 Enterprise View on page 48.

The Job scheduler in Pro2 6.1

The Pro2 Job Scheduler allows users to optionally discontinue the use of operating system specific scheduler utilities like the Windows Task Scheduler. The Pro2 scheduler works across all OpenEdge supported operating system infrastructures and provides a common interface for dealing with complex scheduling requirements. For more information, see How to schedule jobs in Pro2 on page 87.

Status Check improvements in Pro2 6.1

Status check functionality was enhanced to monitor split threads and Enterprise Push program details.

Web user interface enhancements in Pro2 6.1

- The configuration for **LAN/WAN** was moved to the Settings Menu (**Menu** > **Settings** > **LAN/Wan**). This eliminates the perpetual prompting for this information in the New/Edit replication wizard screens.
- **CDC/Trigger update** is now integrated with the **Select Source** page in the **Manage Replication** wizard.
- All watch-box detail grids were modified to include **Clear All** functionality.
- Instance Dashboard improvements:

  The dashboard and menu structure were modified/reorganized to provide consistency with the Enterprise View dashboard. Additional performance graphs and CDC statistics are now available on the dashboard.
- Safari and Internet Explorer are now included in the list of supported browsers.

Get started with Pro2

OpenEdge Pro2 is a lightweight, configurable tool that provides near real-time replication of an OpenEdge database to Microsoft SQL, Oracle, or another OpenEdge database.

Learn about Pro2

Pro2 supports enterprises in delivering the critical business data needed to support analytical and other business intelligence initiatives by replicating specific tables and fields, databases, or multiple databases. All replications facilitated by Pro2 can take place while your transactional database is up and running, removing connectivity limitations, disruptions, and risk to your normal business operations.

With Pro2's web UI you can intuitively configure and monitor your database replications, create new replication processes, and manage bulk loading. To deliver near real-time replication, Pro2 uses either OpenEdge Change Data Capture (CDC) or trigger-based replication for optimal performance with your OpenEdge database.

How it works

Replication triggers, or CDC, automatically capture all data changes that occur on your source database and send those changes to your target database. This means that data, like tables and rows, are not actually transferred from your source database to target database, what has transferred is a log of the changes on your source database. Information is written to the replication queue (ReplQueue) in the internal repl and Pro2 databases to identify the created, updated, or deleted record. The multi-threaded replication process retrieves the updated record and the queued data in the replication is moved to the target database. Done in near real-time, the I/O operation is optimized with the updated record residing in cache. Any analysis can be run against current data in a replicated database, leaving the transactional database resources to their normal activities, keeping your data accurate and secure. This is how Pro2 remains lightweight while providing an up-to-date copy of your source database.

The replication tables in the repl and Pro2 databases consist of minimal data, indexing, and a single sequence for process control. The repl database contains the first nine replication tables, while the Pro2 database contains the remaining twenty-six. If you business needs require, the nine tables that are associated with the repl database can be embedded directly into the source database. A common use case for type of configuration is if you need to migrate your the data in one OpenEdge database to another in preparation of an upgrade or a dump and load.

To replicate Pro2 uses a replication processor that cycles through replication records periodically, based on user configuration, and replicates the table data directly to the target database via the OpenEdge DataServer. All replication data is captured and maintained in Pro2 replication database tables to facilitate table mapping between source and target database, as well as to maintain the queue record, the record lock, the database connection and configuration details, and the thread level information.

Pro2 allows you to configure the time interval between the replication operations. You can customize the interval in real time according to your needs. For example, you can set the interval to every *n* seconds, hourly, twice a day, or daily depending on your specific business needs.

Additionally, you can pause all replications or a specific table replication temporarily should the need arise. You can restart the paused replication at any time. Keep in mind that, while the replication process is suspended, users of the target database do not have access to the records that have been updated in your source OpenEdge database until the replication process is restarted and the replication queue is completely cleared.

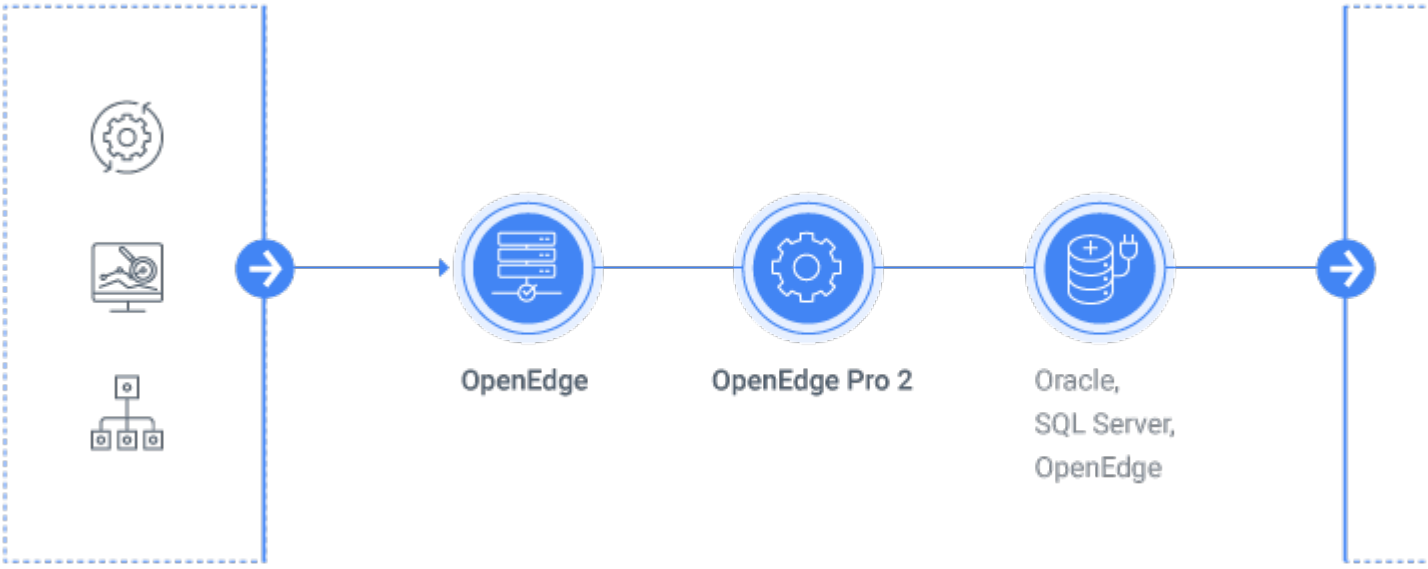
Note: If the replication process is paused, the backlog of data change events could take a long time to process if paused for long period of time.

While the repl and Pro2 databases are lightweight, their initial setup and configuration should be subject to every consideration and precaution as if you were creating a database for an enterprise application in a production environment. Without these databases, data is not replicated from your source database to your target database. This can cause loss of synchronization.

Confer with your database administrator and other key stakeholders to when creating the repl and Pro2 database for your production environment. For more information about database administration, see OpenEdge database essentials.

Architecture

The basis of the replication functionality consists of five primary programs that generate the replication triggers, generate the replication processor functionality, and process the replication records, and a web based user interface application for monitoring and administering the replication processor and configuration.



LAN or WAN

Pro2 architecture varies over local area network (LAN) or wide area network (WAN). Each configuration, whether WAN or LAN, includes:

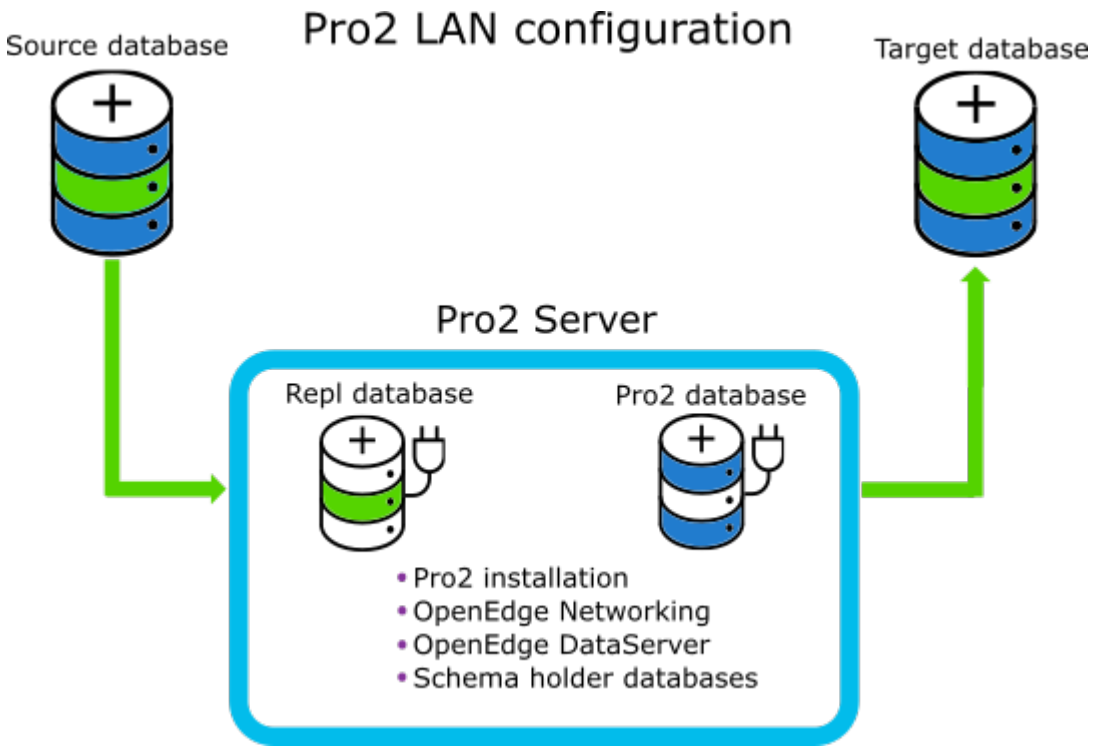
- An OpenEdge source database. There is usually a user application tied to the source database.
- The Pro2 and repl databases.
- A target database that is either a Microsoft SQL Server, Oracle, or an OpeEdge database.

LAN

Pro2 architecture over LAN consists of the following components:

- A source, OpenEdge, database with a user application tied to it.
- The Pro2 and replication databases.
- A target database which can be either Microsoft SQL Server, Oracle or an OpeEdge database.

Note: If your target database is not an OpenEdge database, you need an additional schema holder database which contains information about the data definitions of the target databases.

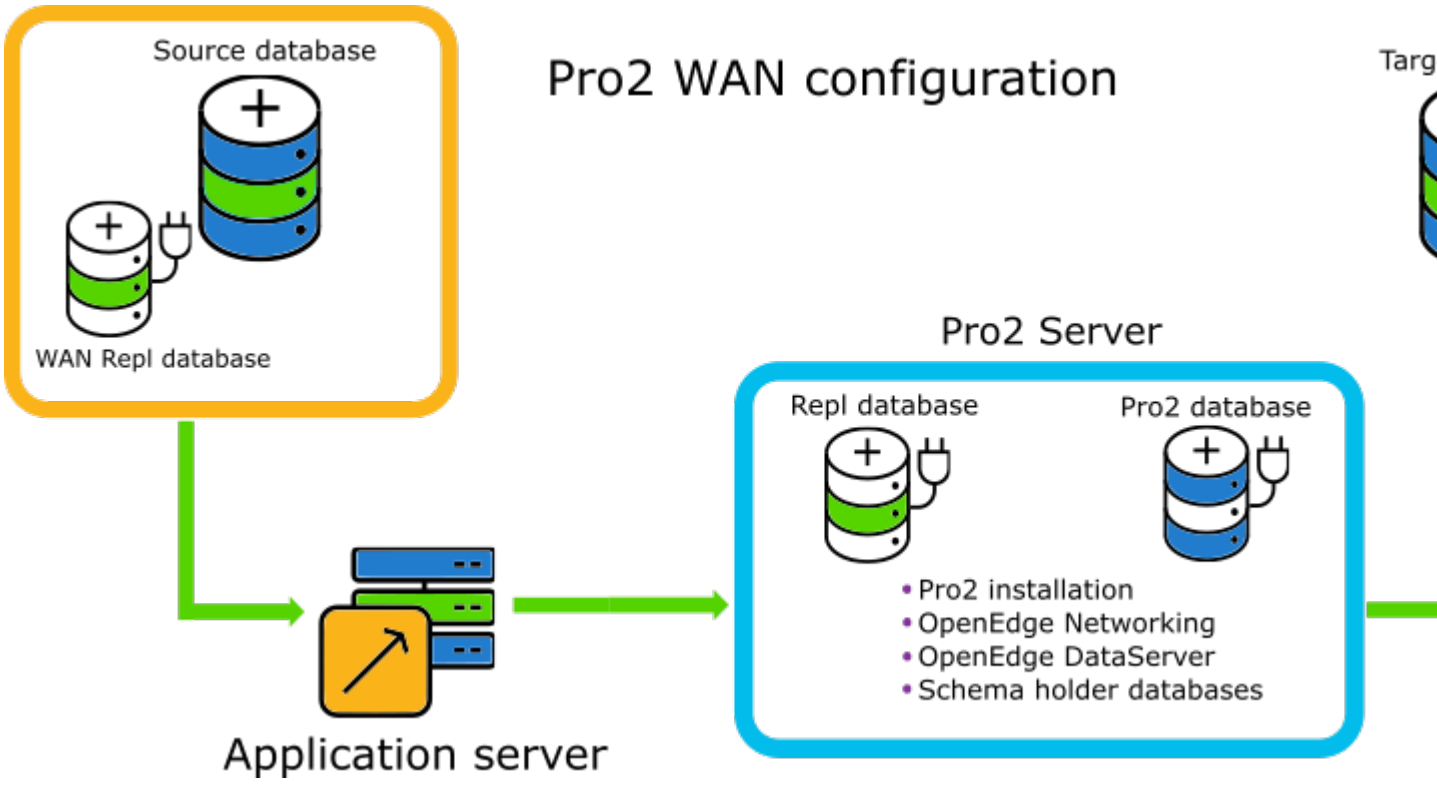


WAN

When the target database is in a different geographic location than the source OpenEdge database, configure Pro2 to include a Progress Application Server on the source database server in addition to the standard configuration on the Pro2 server. In this context, the Pro2 server is where the Pro2 replication instance is configured and operating.

Instead of running the replication procedures using typical client/server mode connections to the source and repl databases, Pro2 sends requests to PAS for OpenEdge (or a classic application server) on the database server which returns data to the calling procedure on the Pro2 server.

Note: The Pro2 server must be on the same local area network as the target database server.



Installation planning and considerations

Before you proceed with the initial installation and configuration of Pro2, take time to create an installation plan for your implementation. Confer with your database administrator, system administrator, and other key stakeholders. Gather information about your implementation needs and consider the following questions.

- What is the size of your OpenEdge database?
- Do you have an environment to test the installation in?
- Where will Pro2 be installed? Keep in mind that it can be installed on its own physical or virtual machine, or on the same machine as your target database.
- Do you require a LAN or WAN configuration?
- Which tables and fields need to be replicated? You can replicate your entire database a portion of it.
- Do yo need to embed the replication tables in your source database?

Pro2 setup process

- Install and configure Pro2 for LAN on page 15
- Install and configure Pro2 for WAN on page 29

Install and configure Pro2 for LAN

A LAN configuration for Pro2 and your OpenEdge application is best suited for on-premises business solutions. For example, assume that you have an OpenEdge application, OpenEdge database, and a Microsoft SQL Server database set up and configured for your on-premises environment.

In this scenario, your OpenEdge database supports your OpenEdge application and you use the Microsoft SQL Server database for reporting and analytics. Rather than periodically updating your Microsoft SQL Server database with information and changes from your OpenEdge database, you can use Pro2 to replicate changes on your OpenEdge database to your Microsoft SQL Server database in near real time. To achieve this, Pro2 uses a series of replication databases, configuration tables, and a queuing table to communicate database changes from the source database to the target database, in this case from your OpenEdge database to your Microsoft SQL Server database. In short, the Pro2 application connects to the source, replication, and target databases, determines what changed on the source database, fetches the records that were changed, and pushes those records to the target database.

You can install Pro2at the same time you install and configure OpenEdge, or you can install Pro2 as an add-on to complement your OpenEdge environment later on. Installing Pro2 as an add-on can be especially useful when you are upgrading OpenEdge versions, or need to migrate databases. For more information about database migration, see Migrate your OpenEdge database with Pro2 on page 83.

Before you can install Pro2, you must first install and configure OpenEdge. This installation must be on a separate machine from your source OpenEdge database installation. It can be on its own machine, virtual or physical, or on the same system as your target database. Likewise, when you install Pro2, you can install and configure it on its own machine or on the same system as your source database. The installation of OpenEdge and Pro2 on its own machine, is sometimes referred to as the Pro2 server. The Pro2 server contains PAS for OpenEdge instance, the Repl and Pro2 databases, and the configuration information for the web based user interface.

System requirements

The specifications for a Pro2 server are generally the same whether server is a physical or virtual machine. Because are determined by the OS being loaded on the box. However the following is a general idea of new specifications.

- Minimum 2 processors.
- Memory: If using a 64-bit operating system, use 8 gigabytes of RAM.

  Note: If your machine also hosts a SQL server, then use a 64-bit platform with a minimum 8 gig of memory.

- Network speed: minimum 1 gigabit per second.
- Disk: 50 gigabytes of allocated space. This allocation is for both the installation and for any content generated by Pro2, such as log files.

  Note: If your machine also hosts a SQL Server database, then the size of the SQL Server database will be approximately twice the size of the Progress data space requirements.

  Note: In either a physical or virtual environment it is recommended to spread the I/O. On a physical machine this means separating the operating system/Pro2 and/or SQL database onto separate physical drives. This is the same for a virtual machine the where “disks” become multi-VMDK file (Virtual Machine Disk). For example separate the operating system and Pro2 installation onto separate VMDK files.

These are minimum recommendations. All parameters can be increased to accommodate more robust hardware.

For more information about OpenEdge system requirements, see the Progress Product Availability Guide.

Before you begin

Ensure that you have OpenEdge installed and configured on the same machine that you are going to install Pro2 on. For more information about installing OpenEdge, see Install OpenEdge on Windows.

Main steps to install and configure Pro2

1. Install Pro2.
2. Create a Repl and Pro2 database.
3. Start the PAS for OpenEdge instance for the Pro2 web user interface.
4. Load the configuration file.
5. Add the source database details.
6. Add the target database details.
7. Generate the target schema file.
8. Load the target database table file.
9. Map the tables in the source database to the target database.
10. Generate the replication code based on your configuration.

Install Pro2

1. Close all applications before beginning the installation.
2. Run the `Pro2.exe` file.

   There can be a numeric value in the file name that indicates the version of Pro2, for example, `Pro2-6.2.exe`.
3. In the Pro2 installation wizard, click **Next** to Choose the install folder.
4. Specify the path, or click **Choose** to browse and select where you want to install Pro2.

   The default installation directory is `C:\Progress\Pro2`.
5. In the **Choose OpenEdge** window, specify the path, or select **Choose** to browse and set the OpenEdge home directory.

   The OpenEdge home directory is where OpenEdge is installed on your machine.

   The default directory is `C:\Progress\OpenEdge`.
6. On the **PAS Ports** screen, enter the relevant PAS for OpenEdge information in the **Port** and **Instance Name** fields.

   The default instance name is Pro2Web. You can change this name to according to your needs. It is recommended to change the instance name for each Pro2 installation.

   Note: For each installation of Pro2, you must specify different port information. No two installations can use the same port credentials on the same machine.

   Click **Next** to view the **Choose Shortcut Folder** window.
7. On the **Choose Shortcut Folder** window, choose where you want to create a shortcut for Pro2.

Note: Select the **Create Icons for All Users** option only if **In a new Program Group**, **In the Start Menu**, or **On the Desktop** is selected.

8. Click **Next** and review the options you chose for this installation in the **Pre-Installation Summary** page.
9. Click **Install**.

Note: The Pro2 installation wizard can take several minutes to complete and may appear to lag at times. Allow the installation wizard to complete. Do not prematurely cancel the process.

10. Click **Done**.

After the installation is complete, go to the Pro2 root folder to continue the configuration process. If you used the default settings during the installation, then this folder is `C:\Progress\Pro2`.

All files, scripts, and procedures for Pro2 are relative to the root folder. This allows Pro2 to be installed several times on the same drive without each installation interfering with each other. This functionality was designed to allow you to be able to create development and production environments to test implementations, as needed, without affecting your production environment.

For more information about the Pro2 root folder and file directory, see the Directory reference at the end of this guide.

Create a Repl and Pro2 database

In this part of the configuration process, you build a Repl and Pro2 database by using Proenv commands and the `repl.df` and `pro2.df` files.

Standard OpenEdge database utilities are used to create the Repl database. The structure file (`.st`) and schema definition file (`.df`) used to create the Repl database can be found in the `db` folder of the Pro2 root folder. If you are using replication triggers with a LAN configuration, all configuration files, parameter (`.pf`) files, and scripts need to be modified to include a connection to the Repl database wherever there is a connection made to the source database. Change data capture (CDC) is not affected by this requirement.

It is important to note that this procedure describes the high level steps to create the Repl and Pro2 databases. When creating these databases every consideration and precaution should be taken, as if you were creating a database for an enterprise application in a production environment, to ensure their availability. Without these databases, data is not replicated from your source database to your target database. This can cause loss of synchronization.

Confer with your database administrator and other key stakeholders to when creating the Repl and Pro2 database for your production environment. For more information about database administration, see OpenEdge database essentials.

To create a Repl and Pro2 database:

1. Open a **Proenv** instance.
2. Set your working directory to the `db` folder in the Pro2 root directory.

Note: Check the directory of the `db` folder to view the database instances. Type `dir` after navigating to the database folder.

3. Create an empty repl database instance by typing the following command `prodb repl empty`.
4. Create an empty Pro2 database instance by typing the following command `prodb pro2 empty`.

Note: Ensure that the you set the code page of the Repl and Pro2 databases to match the code page of your source and target databases.

5. Load the table definitions file in the Repl database by typing `prowin repl -1`.

   The **Procedure Editor** window appears.

6. Navigate to **Tools** > **Data Administration**.
7. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)…**.
8. In the **Load Data Definitions** window, load the `repl.df` file and click **OK**.

Note: The `repl.df` file is located in the `db` folder of the Pro2 root folder.

9. Load table definitions in the Pro2 database by entering `prowin pro2 -1`.

   The **Procedure Editor** window appears.

10. Navigate to **Tools** > **Data Administration**.
11. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**.
12. In the **Load Data Definitions** window, load the `pro2.df` file and click **OK**.

Note: The `pro2.df` file is located in *`Pro2 folder`*`/db`.

13. Start the repl and pro2 databases by typing:
    - `Proserve repl`
    - `Proserve pro2`
14. Add the Pro2 database connection details to the `replProc.pf` file in the Pro2 root folder under `\bprepl\Scripts\replProc.pf` to connect the replication threads and PAS for OpenEdge instance to the Pro2 database. Navigate to the `bprepl\scripts` folder and open the `replProc.pf` file. Add the complete path of the database in the file and click **Save**.
15. To check both Repl and Pro2 database instances are connected successfully, open the **Procedure Editor** from the `bprepl\Scripts` folder and navigate to **Tools** > **Data Dictionary**. If the connection is successful, the **Tables** section displays the tables.

Start the PAS for OpenEdge instance and log in to the Pro2 web application

Now that the database instances are set up successfully, start the PAS OpenEdge instance and log in to the Pro2 web application.

1. Start a Proenv instance.
2. Navigate to the **Pro2** folder by entering `cd` followed by name of the folder where you installed Pro2, for example, `cd Pro2v62`.
3. Enter *`instance name`*`\bin\tcman.bat env`.

Note: The default instance name is Pro2Web.

4. Enter *instance name*\bin\tcman.bat start.
5. To verify that the instance started successfully, type *instance name*\bin\tcman.bat env. The server running section shows a number if the instance started.
6. Open any browser and log in at localhost:*port number*/pro2/static/ using the default credentials.

Load the configuration file

Before you set up the source and target database connections in the user interface, you must set up a basic configuration by loading the default configuration file. The configuration file consists of all the default properties and records that are necessary to set up the Pro2 environment.

Note: If you use Internet Explorer to load the configuration file, you may encounter errors. To avoid them, use an alternative web browser like Google Chrome.

To load the configuration file:

1. In the Pro2 web interface, navigate to **Actions** > **Tools** > **Load Configuration**.
2. Click **Select Files** and load the .ini configuration file, for example replbasev610.ini, from the Pro2 folder.
3. Click **Submit**. A success or failure notification is displayed after the upload is complete.

You can view the default properties and records on the **Properties** tab after the configuration file is successfully loaded.

Start the job runner

To start the job runner and initiate the scheduled Enterprise Push job:

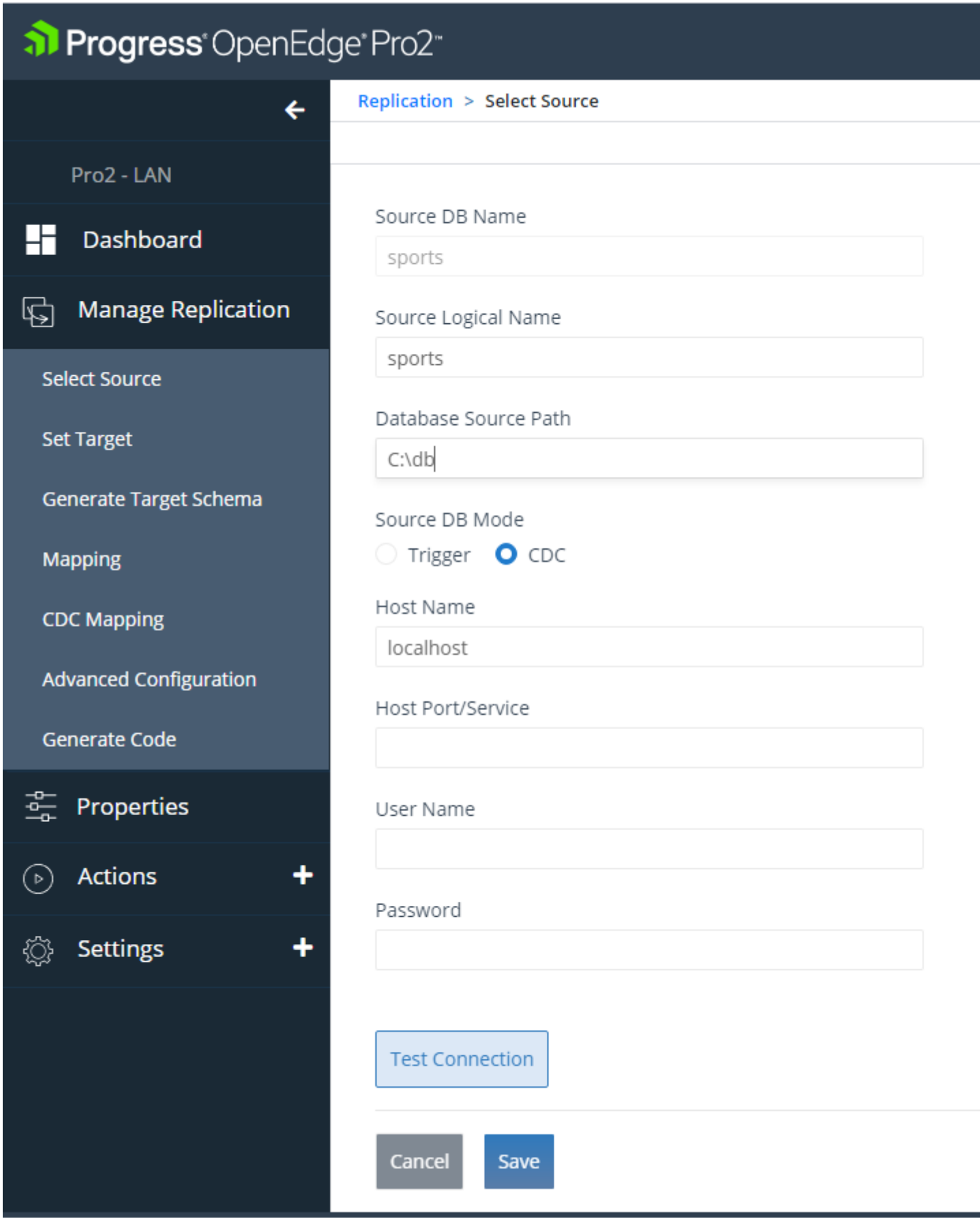
1. Navigate to the Pro2 Scripts folder, for example C:\Pro2\bprepl\Scripts.
2. Select and execute the jobrunner.bat file.
3. Navigate back to the web interface and confirm that the jobrunner.bat file is running by checking the status in the Pending Jobs watch-box.

Connect to a source and target database

The Pro2 source database is an OpenEdge database where the data modifications are made, and the target database is the database where the modifications are replicated. The source database must be an OpenEdge database, the target database can be an OpenEdge database, Microsoft SQL Server, or an Oracle database.

Add the source database details

You can set up either trigger-based replication or CDC-based replication. In a trigger-based-replication, the replication queue records (also known as ReplQueue records) are generated by deploying Pro2 triggers on a source database. In a CDC-based replication the ReplQueue records are generated from change tracking table. However, for CDC-based replication, the source database must be CDC enabled.


Progress OpenEdge Pro2
Replication > Select Source
Pro2 - LAN
Dashboard
Manage Replication
Select Source
Set Target
Generate Target Schema
Mapping
CDC Mapping
Advanced Configuration
Generate Code
Properties
Actions
Settings
Source DB Name
sports
Source Logical Name
sports
Database Source Path
C:\db
Source DB Mode
Trigger
CDC
Host Name
localhost
Host Port/Service
User Name
Password
Test Connection
Cancel
Save

To add source database details:

1. From the Pro2 web interface, navigate to the **Manage Replication** tab.
2. Click **New**.

   The **Create Replication** window appears.
3. Set up settings for the replication instance, for example, **Source DB Mode** is set to **Triggers**, and the **Source DB Connection** is set to **LAN**.
4. Click **Next**.

   The **Select Source** tab appears.
5. Enter the name of the source database.
6. Enter the host name.
7. Enter the port number in the `Host Port/Service` field (for example, 2233).
8. (Optional) Enter your user name and password.

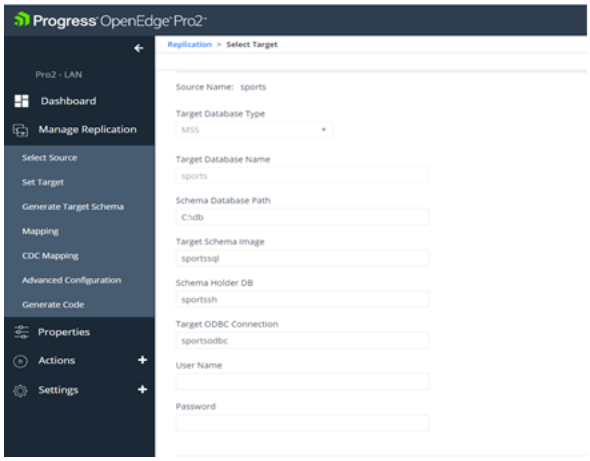
   Note: Enter your user name and password only if it is required by your source database.
9. Click **Test Connection**.

   If the test is successful, move on to set the target database. Review your source database details to ensure that they are correct.
10. Click **Next**.

    The **Set Target** tab appears.

Add the target database details

The target database can be either a Microsoft SQL Server, Oracle database, or OpenEdge database. If your target database is an OpenEdge database, the you can forgo some of the configuration processes like generating the schema holder. However, there for a Microsoft SQL Server or Oracle database there are several configuration considerations to take into account. For more information about configuring Microsoft SQL Server or Oracle as a target database, see Configure Pro2 to target SQL database connectivity on page 69.



To add target database details:

1. Select **Target Database Type**. Choose Microsoft SQL Server, Oracle database, or OpenEdge database.
2. Following fields are automatically filled, but they can be changed if necessary

- **Target Database Name**
- **Target Schema Image**
- **Schema Holder DB**
- **Target ODBC Connection**

3. Enter the source database path. By entering the full path or relative path then there is no need to enter the host and port details.
4. (Optional) Enter your user name and password.

Note: Enter user name and password only if it is required by your target database.

5. Click **Next**.

   The **Generate Target Schema** tab appears.

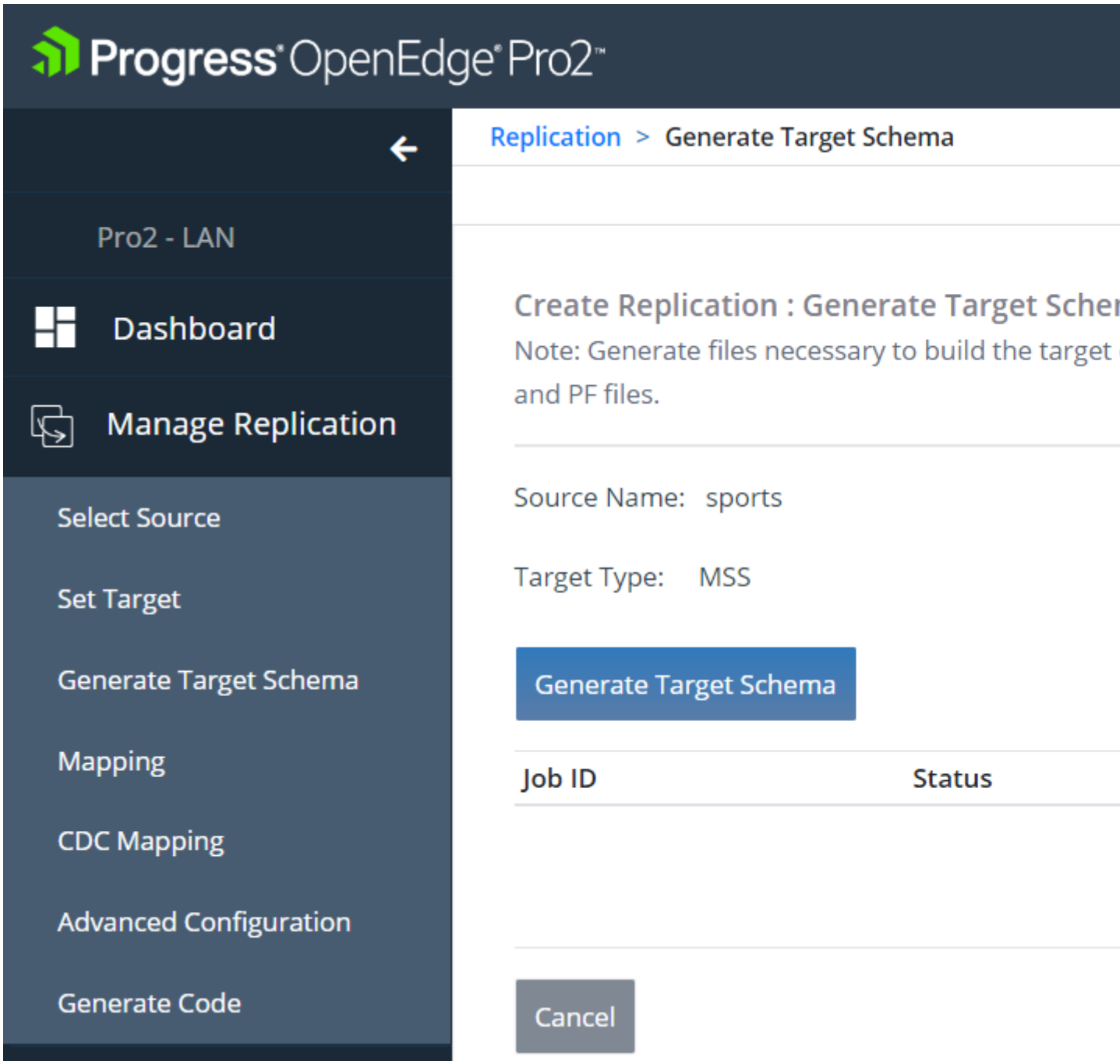
Note: At this point in the process the `replproc.pf` contains the details for your database's .pf file in addition to automatically creating the .pf file.

Here is an example of a .pf file that was created using the sports database and Microsoft SQL Server:

```
# Schema Holder Databases go here.
# If not built yet then just comment out all lines. (Required for MSS/ORA/PROGRESS as target type)
-db C:\web_pro2\pro2\db\sportssh -ld sportssh -RO

# Schema Images go here.
# If not imported yet, then just comment out all lines.
  -db sportsodbc            # Required MSS/ORA
  -ld sportssql                 # Required MSS/ORA

    # Required MSS/ORA value should be (MSS/ORACLE)
    # For Oracle Target type need to enter username and password

  -dt MSS -U sa -P sa
  -Dsrv QT_CACHE_SIZE,30000           #Required for MSS and ORA target type in other type just comment it
  -Dsrv TXN_ISOLATION,1               #Required for MSS target type in other type just comment it
# -Dsrv BINDING,0                     #Removed for v11
  -Dsrv AUTOCOMMIT,1                  #Required for MSS target type only
  -Dsrv PRGRS_PREPCACHE,100           #Required for MSS target type only
  -Dsrv MSS_PRESERVE_CURS,1           #Required for MSS target type only
  -Dsrv PRGRS_LOCK_ERRORS,08501       #Required for MSS target type only
  -Dsrv logginglevel,0,svub,1         #This will reduce the DataServer Log output Required for ORA/MSS

    #Added below to release locks earlier
  -Dsrv PRGRS_NATIVE_LOCKWAIT,0       #Required for MSS target type only
  -Dsrv PRGRS_NOWAIT_OVERRIDE,1       #Required for MSS target type only
```

Generate the target schema file

By generating the target schema file, you load the target database with files necessary to build out the target database. Pro2 reads the source database schema, and generates scripts that are then executed on the target database server to create the target schema. By default, all tables and fields are created in the target. The rest of the files (`drop.sql`, `index.sql`, `trunc.sql`, `errors.log`, and `warnings.log`) are created separately. You can execute these files as necessary. Typically, tables that are not replicated remain unmapped.



To generate target schema file:

1. Select one or more of the following checkboxes:

Note: These options are available only when the target database is an OpenEdge database. If the target database is Microsoft SQL Server or Oracle database, these options are not available.

- **Preserve Area**: Preserves the source database data definition, and copies them to the target database.
- **Generate Pro2 Target Fields**: Generates Pro2-specific target fields in your target database. Doing so avoids unnecessary replications of the source schemas.

- **Generate Index as Inactive**: Deactivates the existing active index.

2. Click **Generate Target Schema**. A table that contains a replication job appears.

Note: Do not execute this job until you run the job runner file. A job runner batch file continuously processes replication jobs from the job queue. This file is in the `Scripts` folder.

---

3. Click **Download** under the **Download Schema files** column, to download the target schema files.

Create a target database

If you have not done so already, create a target database for you replicated data. The Target database can be either a Microsoft SQL server, Oracle, or OpenEdge database. Creating a target database will depend largely on they type of database you want to use and your development tooling. The following procedure lists the general steps to creating your target database.

1. In your database admin tool, like Microsoft SQL server Management Studio, create a new database and name it according to your needs.
2. Create a new database query, and load the schema file into the query.

Create a schema holder for your target database

In this step, you create an empty database as a schema holder for your target database. Your target database can be either Microsoft SQL Server or Oracle. The schema holder is a layer that allows ABL to interact with the Microsoft SQL Server or Oracle database.

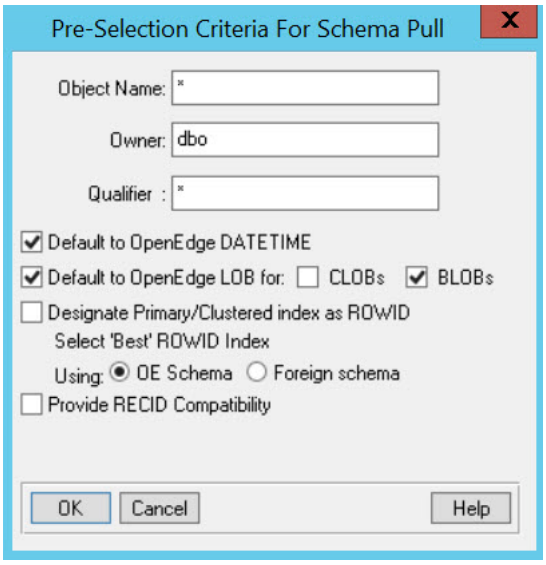
To create a schema holder:

1. Open a **Proenv** window, and type `prodb` *`database name`* `empty` to create an empty schema holder database.
2. Start a single user session by typing `prowin` *`database name`* `-1`.

   The **Procedure Editor** window appears.
3. From the **Procedure Editor**, select **Tools** > **Data Administration**.
4. Select **DataServer** > ***`database type`*** **Utilities** > **Create DataServer Schema** .

   The **Create/Modify Record for Schema** window appears.
5. Specify **Logical Database Name**.
6. If the source database is a SQL server, specify the **ODBC Data Source Name**. Click **OK**.
7. Enter the login ID and password if required in the next dialog box and click **OK**.
8. Enter criteria for **Object Name**, **Object Owner** (typically, `dbo`), and **Qualifier**.
9. Select the **Default to OpenEdge DATETIME** option.
10. If you are using LOB fields in replication, ensure that the **Default to OpenEdge LOB** for **BLOBs** option is selected along with the CLOBs option. This forces the data server to automatically consider certain target side data types to be treated as LOBs.

    By selecting CLOBs, `varchar(max)/nvarchar(max)` data types are automatically converted to CLOBs. By selecting BLOBs the data server is forced to consider all `varbinary(max)` as BLOBs. On source fields that are character types, Pro2 may convert some types to `varchar(max)` when the source width is wider than the `MAX_CHAR_WIDTH` property setting. This conversion has the added benefit of avoiding row size limits for SQL Servers. However, `varchar(max)/nvarchar(max)` are not necessarily treated as CLOBs.



11. Click **OK** to finish.

Load the target database table definitions

After the target schema file is downloaded, load the target database with the appropriate tables required for replication.

For OpenEdge target databases:

1. Cut and paste the target schema file into the `db` folder of Pro2.
2. Open a **Proenv** window, type `prowin -db` target and press **Enter**.
3. In the Procedure Editor, select **Tools** > **Data Administration**.
4. Select **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**
5. Add the target schema file as the **Input File** and click **OK**.
6. Navigate to the Pro2 web interface and click **Next**.

   The **Mapping Source to Target Tables** tab appears.

For Microsoft SQL Server or Oracle target databases:

1. Open a **Proenv** window, type `prowin` *`database name`* `-1` and press **Enter**.

   The **Procedure Editor** window appears.

2. Select **Tools** > **Data Administration**.
3. Select **Database** > **Select Working Database**.
4. Click **OK**. The **Data Administration** window appears.
5. Select **DataServer** > ***`database type`*** **Utilities** > **Update/Add Table Definitions…**

   The **Pre-Selection Criteria for Schema Pull** window appears.

6. Enter `dbo` for the **Owner**, select the **Default to OpenEdge DATETIME** and **Default to OpenEdge LOB for: CLOBs and BLOBs** checkboxes. Click **OK**.

   The **Select *`database type`* Objects** window appears.

7. Click **Select Some...**.

   The **Select Objects by Pattern Match** window appears.

8. Enter `dbo` for the **Owner**, `table` as **Object Type**, and click **OK**.
9. Truncate the *`database name`* database to restrict operations in this database. This database should be a read-only database. Enter the following command in the **Proenv** window.

   `proutil` *`database name`* `-C truncate bi`

10. Navigate to the Pro2 web interface and click **Next**.

    The **Mapping Source to Target Tables** tab appears.

Map tables and fields

You must map data fields between your source and target database so that data in the source can be replicated to the target. You can do this from the **Mapping** dashboard in the web interface. The Repl database contains all the mapping information for the replication of the databases, tables, and fields between the source and target. This information is stored in the `ReplDBXref, ReplTableXref,` and `ReplFieldXref` tables. Mapping information can be saved to and loaded from a text file.

Depending on the number of schemas, primary keys, and foreign keys of the databases, the data sources and the database mapping information has varying degrees of complexity.

Map source tables to target tables and generate code

To map source tables to target tables and generate code:

1. On the **Mapping Source to Target Tables** tab, select the tables from the source database that you want to map with the target database. You can map the tables in the following ways:

   - Map—Choose the **Map** option if you want to map a single source table with a single target table.
   - Automap—Choose the **Automap** option if you want to map one or more source tables to the corresponding target tables.

   Note: You can also unmap the tables based on your needs using the **Unmap** option.

2. Select the tables to map, and click **Automap** click **Next**.
3. In the **Advanced Configuration** tab, change thread assignments and other optional control flags if required.

Note: The **Advanced Configuration** tab is an optional settings tab.

4. Click **Next**.

   The **Generate Code** tab appears.

5. Click **Generate Code**. You are directed to the Pro2 web interface dashboard. Click the **Pending Jobs** watch-box to see which replication jobs are running.
6. Run the replication file to execute the replication jobs. In **Windows Explorer**, navigate to `\bprepl\Scripts`.
7. Double-click the **Replbatch1.bat** file.

   The corresponding replication thread starts running on the dashboard.

Note: In a CDC-based replication, double-click the **CDCbatch.bat** file.

8. Refresh the Pro2 user interface to complete the replication configuration process.

Manually test a replication

Testing is an important part of any enterprise applications configuration process. Consider an testing example where a city is assigned to a customer. This change is made in the source database, and needs to be replicated to the target database. To perform the test scenario:

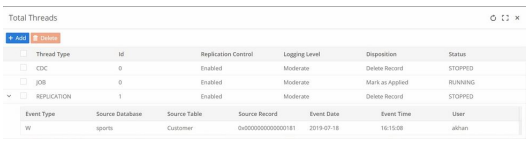
1. In a **Proenv** window, navigate to your Pro2 root folder. For example, `C:\Pro2v62\db`.
2. Enter the command `prowin -db` *`database name`* `-db repl`.

   The **Procedure Editor** window appears.

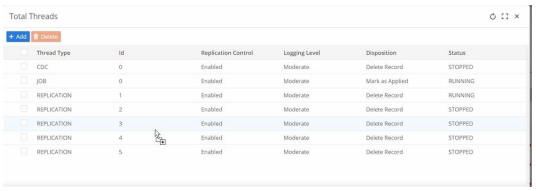
3. Assign city to the first customer in source database.

```
FIND FIRST customer
ASSIGN city = “NY”
```

4. Open the Pro2 web interface and click the **Total Threads** watch-box. Notice the event type in the **Replication thread** window. This event will show the recent change made to the source database.



5. Run the replication file by navigating to `\bprepl\Scripts` .
6. Double-click the **Replbatch1.bat** file to run the replication thread. This action removes the event type in the thread which simulates that the replication record was processed by the thread and sent to the target database.



Directory reference

The following is a description of the folders under the **Pro2** root folder.

Folder	Description
`bprepl`	Root directory for application programs
`bprepl\AppSrv`	Used in WAN implementations
`bprepl\images`	Image files that can be used for shortcut icons
`bprepl\misc`	Miscellaneous replication files and procedures
`bprepl\repl_as_tgt`	Used in WAN implementations
`bprepl\repl_d`	Directory for the Pro2 generated database replication delete trigger procedures
`bprepl\repl_export`	Used in WAN bulk loads
`bprepl\repl_inc`	Directory for Pro2 generated assigned include files
`bprepl\repl_jtrig`	Directory for any Java triggers
`bprepl\repl_log`	Default location of log files
`bprepl\repl_mgtrig`	Directory of tables that require merged triggers
`bprepl\repl_mproc`	Directory for the Pro2-generated bulk copy procedures
`bprepl\repl_proc`	Directory for the Pro2-generated replication library
`bprepl\repl_w`	Directory for Pro2-generated database replication write trigger procedures
`bprepl\repl_tmpl`	Directory containing the templates used for various code generation
`bprepl\Scripts`	Directory containing the `.pf` files, scripts, and shortcuts to start the Pro2 Admin Tool, replication processors, and bulk load procedures
`bprepl\SQL_inc`	Directory for the direct, SQL assigned include files
`bprepl\SQL_mproc`	Directory for the direct, SQL bulk copy procedures
`bprepl\SQL_proc`	Directory for the direct, SQL replication procedures

Folder	Description
`custom`	Directory to deploy your customized code so that it overrides the existing code in the `bprepl` folder **Note:** This folder must maintain the `bprepl` folder structure and is applicable for both LAN and WAN configurations
`db`	Location of the schema holder database. Also, the initial temporary location for Repl database during implementation
`Installation`	Initially empty. Used to save installation site-specific files
`misc`	Miscellaneous Pro2 utilities
`tmp`	Miscellaneous Pro2 temporary files used during implementation

ReplProperties and ReplQueue

The OpenEdge replication source database and target database tables are stored in property files. There are configuration settings whose values you set in these tables that control different aspects of your replication environment. The following are the tables for the Pro2 replication environment:

- `ReplProperties`—Configuration settings such as log file location, logical delete tables, and specification of procedure templates are stored in the `ReplProperties` table. Configuration settings can be saved to and loaded from a text file from the **New** button in the **Manage Replication** window on the Pro2 web interface.
- `ReplQueue`—Information about change events is stored in the `ReplQueue` table. This information includes the `ROWID` of the record changed, event date, time, and queue thread. Typically, repl queue records represent updates made to the source database that are waiting to be written to the target SQL database.

Install and configure Pro2 for WAN

A WAN configuration for Pro2 and your OpenEdge application is best suited for cloud-based business solutions across multiple geographies. For example, assume that you have OpenEdge applications and databases spread across the US, Europe, and Asia, with a consolidated Microsoft SQL Server database in the US that is used for reporting and analytics. In this configuration, the source databases spread across the globe communicate with an application server that in turn, sends data to the target Microsoft SQL Server database.

You can install Pro2 along side of OpenEdge during your initial setup and configuration, or you can install Pro2 as an add-on to complement your OpenEdge environment later on. Installing Pro2 as an add-on can be especially useful when you are upgrading OpenEdge versions, or need to migrate databases. For more information about database migration, see Migrate your OpenEdge database with Pro2 on page 83.

If you are installing Pro2 along with OpenEdge, then you will need to first install and configure OpenEdge, after the installation and configuration is complete, you then install and configure Pro2.

The Pro2 WAN configuration setup differs from the LAN configuration process in that, after the initial installation of Pro2, you must configure the application server before you can set replication configurations for your Pro2 instances. Moreover, unlike the LAN configuration, the source database does not directly communicate changes to the replication and target databases. The source database is connected to a Progress Application Server instance, which is then connected to an empty synchronized-copy of the replication database that connects to the Pro2 facing replication database.

System requirements

The specifications for a Pro2 server are generally the same whether server is a physical or virtual machine. Because are determined by the OS being loaded on the box. However the following is a general idea of new specifications.

- Minimum 2 processors.
- Memory: If using a 64-bit operating system, use 8 gigabytes of RAM.

  Note: If your machine also hosts a SQL server, then use a 64-bit platform with a minimum 8 gig of memory.

- Network speed: minimum 1 gigabit per second.
- Disk: 50 gigabytes of allocated space. This allocation is for both the installation and for any content generated by Pro2, such as log files.

  Note: If your machine also hosts a SQL Server database, then the size of the SQL Server database will be approximately twice the size of the Progress data space requirements.

  Note: In either a physical or virtual environment it is recommended to spread the I/O. On a physical machine this means separating the operating system/Pro2 and/or SQL database onto separate physical drives. This is the same for a virtual machine the where “disks” become multi-VMDK file (Virtual Machine Disk). For example separate the operating system and Pro2 installation onto separate VMDK files.

These are minimum recommendations. All parameters can be increased to accommodate more robust hardware.

For more information about OpenEdge system requirements, see the Progress Product Availability Guide.

Before you begin

Ensure that you have OpenEdge installed and configured on the same machine as Pro2. For more information about installing OpenEdge, see Install OpenEdge on Windows.

Main steps to install and configure Pro2

1. Install Pro2 by using the installation wizard.
2. Create and configure a WAN side repl database.
3. Create and configure the WAN application server.
4. Create a Pro2 side repl and Pro2 database.
5. Create an empty local copy of the source database.
6. Start the PAS for OpenEdge instance for the Pro2 web user interface.
7. Load the configuration file.
8. Connect to your WAN application server instance.
9. Start the job runner.
10. Add the source database details.
11. Generate and load the source schema file.
12. Add the target database details.
13. Create the target database.
14. Create a schema holder for your target database.
15. Generate the target schema file.
16. Load the target database table file.
17. Map the tables in the source database to the target database.
18. Generate the replication code based on your configuration.

Install Pro2

1. Close all applications before beginning the installation.
2. Run the `Pro2.exe` file.

   There can be a numeric value in the file name that indicates the version of Pro2, for example, `Pro2-6.2.exe`.
3. In the Pro2 installation wizard, click **Next**. The **Choose the Install Folder**screen appears.
4. Specify the path or click **Choose** to browse and select where you want to install Pro2.

   The default installation directory is `C:\Progress\Pro2`.
5. In the **Choose OpenEdge** window, specify the path, or select **Choose** to browse and set the OpenEdge home directory.

   The OpenEdge home directory is where OpenEdge is installed on your machine.

   The default directory is `C:\Progress\OpenEdge`.
6. On the **PAS Ports** screen, enter the relevant PAS for OpenEdge information in the **Port** and **Instance Name** fields.

   The default instance name is Pro2Web. You can change this name to according to your needs. It is recommended to change the instance name for each Pro2 installation.

   Note: For each installation of Pro2, you must specify different port information. No two installations can use the same port credentials on the same machine.

   Click **Next** to view the **Choose Shortcut Folder** window.

7. On the **Choose Shortcut Folder** window, choose where you want to create a shortcut for Pro2.
8. Click **Next** and review the options you chose for this installation on the **Pre-Installation Summary** page.
9. Click **Install** to continue with the installation process.

Note: The Pro2 installation wizard can take several minutes to complete and may appear to lag at times. Allow the installation wizard to complete before prematurely canceling the process.

10. Click **Done** to finish the installation process.

After the installation is complete, go to the Pro2 AS_Src root folder to continue the configuration process. If you used the default settings during the installation, then this folder would be `C:\Progress\Pro2\bprelp\AppSrv\AS_Src`. Copy this folder to the machine where you have your source database, and rename the folder.

All files, scripts, and procedures for Pro2 are relative to the root folder. This allows for Pro2 to be installed several times on the same drive without each installation interfering with each other. This functionality was designed to allow you the be able to create development and production environments to test implementations as needed without the risk affecting your production environment.

Create and configure a Repl database for WAN

On this step of the configuration process you create a WAN side Repl database. This Repl database is hosted on the same machine as your source database and communicates directly with the source database and the WAN application server.

It is important to note that this procedure describes the high level steps to create the WAN side Repl database. Later in the configuration process, you will need to create another Repl database on the same machine as your Pro2 database and Pro2 installation. When creating these databases every consideration and precaution should be taken, as if you were creating a database for an enterprise application in a production environment, to ensure their availability. Without these databases, data is not replicated from your source database to your target database. This can cause loss of synchronization.

1. On you source database machine, where you have just copied the AS_Src root folder to, open a **Proenv** instance.
2. Set your working directory to the db folder in the WAN Pro2 root directory.

   Note: Check the directory of the db folder to view the database instances. Type `dir` after navigating to the database folder.

3. Create an empty Repl database instance by entering the following command `prodb repl empty`.

   Note: Ensure that the you set the code page of the Repl database to match the code page of your source and target databases.

4. Start the Repl database with the `proserv repl` command.
5. In the WAN Pro2 root directory, go the scripts folder `bprepl\scripts` and add the application server connection details to the `AppSrv.pf` file to connect the replication threads and PAS for OpenEdge or classic application server instance to the Repl database and your source database. Add the complete path of the databases in the file and click **Save**.
6. Load the table definitions file in the Repl database by first entering `prowin repl -1`.

   The **Procedure Editor** window appears.

7. Navigate to **Tools** > **Data Administration**.
8. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)…**.
9. In the **Load Data Definitions** window, load the `repl.df` file and click **OK**.

   Note: The `repl.df` file is located in *`Pro2 folder`*`/db`.

10. Start the Repl database by entering: `Proserve repl`
11. To confirm that the Repl database is connected, open the **Procedure Editor** from the `bprepl\Scripts` folder and navigate to **Tools** > **Data Dictionary**. If the connection is successful, the **Tables** section displays the tables.

Create and configure a WAN application server

Creating your WAN application server can vary widely depending on your specific business needs and application environment. In general, creating a PAS for OpenEdge instance involves defining a name and location for the instance as well as defining various configuration properties. When you create an instance, the properties of the instance are automatically set to the directory where the new instance is created and the related files and logs are stored. The following properties are set when the new instance is created and can be controlled by the system administrator:

- The environment variables and paths for running commands
- The HTTP and HTTPS transports, and ports
- The username and password for the instance
- The type of instance (development or production)

For more information about PAS for OpenEdge, see Learn about PAS for OpenEdge administration.

To create and configure a WAN application server:

1. On your source database machine, open a **Proenv** instance.
2. Use the `pasman create` command to create an application server instants. For example

   ```
   pasman create -p 11000 -P 11001 -s 11002 P2WAN
   ```

   - On Windows, a shutdown port, set by the `-s` option, is required.
   - The `-p` (lower case) option specifies the HTTP port. The default is 8810.
   - The `-P` (upper case) option specifies the HTTPS port. The default is 8811.
   - In this example, the instance-path or instance-name is P2WAN
3. Go to the instance-path for your application server using the `cd` command, for example:

   ```
   cd P2WAN
   ```
4. (Optional) Add OpenEdge Management to your instance if your business needs require, for example:

   ```
   bin\tcman.bat deploy C:\Progress\OpenEdge\servers\pasoe\extras\oemanager.war
   ```
5. (Optional) Add the PAS for OpenEdge Health scanner to your instance if you want to use it health monitoring capabilities, for example:

   ```
   bin\tcman.bat deploy C:\Progress\OpenEdge\servers\pasoe\extras\oehealth.war
   ```
6. Use the `oeprop` command to set the application server project directory to the WAN root folder, for example:

   ```
   bin\oeprop AppServer.Agent.PROPATH="C:\Pro2Wan_Scr,
   C:\Pro2Wan_Scr\pbrepl,${CATALINA_BASE}/OPENEDGE,${DLC}/tty,${DLC}/tty/netlib/OpenEdge.Net.pl"
   ```
7. Use the `bin` command to set the work directory to the WAN root folder, for example:

   ```
   bin\oeprop AppServer.Agent.workDir="C:\Pro2Wan_Src"
   ```
8. Set the instance-path project directory by using the `oeprop` command, for example:

   ```
   bin\oeprop
   AppServer.Agent.P2WAN.PROPATH="C:\Pro2wan_Src,C:\Pro2wan_Src\bprepl,${CATALINA_BASE}/webapps/pro2/WEB-INF/openedge,${CATALINA_BASE}"
   ```
9. Set the start up parameters by using the `oeprop` command to point to the `AppSrv.pf` file:

   ```
   bin\oeprop AppServer.SessMgr.P2WAN.agentStartupParam="-T C:\Pro2Wan-Src\tmp -pf
   bprepl\scripts\AppSrv.pf"
   ```
10. Designate which application server adapters are to be enabled are to be enabled, for example:

    ```
    bin\oeprop P2WAN.ROOT.APSV.adapterEnabled=1
    bin\oeprop P2WAN.ROOT.REST.adapterEnabled=0
    bin\oeprop P2WAN.ROOT.SOAP.adapterEnabled=0
    bin\oeprop P2WAN.ROOT.WEB.adapterEnabled=0
    ```
11. Test your PAS for OpenEdge instance with the `pasman test -I` *`instance name`* command, for example:

    ```
    pasman test -I P2WAN
    ```
12. After the test is successfully completed, use the `pasman pasoestart -restart -I` *`instance name`* command to restart your instance for example:

    ```
    pasman pasoestart -restart -I P2WAN
    ```

For more information about the OpenEdge properties command (oeprop), see OEPROP.

Create a Repl and Pro2 database:

While similar to the Repl database that you created earlier, this Repl database is hosted on the same machine as your Pro2 installation. You build a Repl database by using Proenv commands and the `repl.df` file.

Before you create your replication database, navigate back to the machine where you have Pro2 installed. This machine should be different than your source database machine where your PAS for OpenEdge instance is configured.

Standard OpenEdge database utilities are used to create the Repl database. The structure file (`.st`) and schema definition file (`.df`) used to create the Repl database can be found in the `db` folder of the Pro2 root folder. If you are using replication triggers with a LAN configuration, all configuration files, parameter (`.pf`) files, and scripts need to be modified to include a connection to the Repl database wherever there is a connection made to the source database. Change data capture (CDC) is not affected by this requirement.

It is important to note that this procedure describes the high level steps to create the Repl and Pro2 databases. When creating these databases every consideration and precaution should be taken, as if you were creating a database for an enterprise application in a production environment, to ensure their availability. Without these databases, data is not replicated from your source database to your target database. This can cause loss of synchronization.

Confer with your database administrator and other key stakeholders to when creating the Repl and Pro2 database for your production environment. For more information about database administration, see OpenEdge database essentials.

To create a Repl and Pro2 database:

1. Open a **Proenv** instance.
2. Set your working directory to the `db` folder in the Pro2 root directory.

   Note: Check the directory of the `db` folder to view the database instances. Type `dir` after navigating to the database folder.

3. Create an empty repl database instance by typing the following command `prodb repl empty`.
4. Create an empty Pro2 database instance by typing the following command `prodb pro2 empty`.

   Note: Ensure that the you set the code page of the Repl and Pro2 databases to match the code page of your source and target databases.

5. Load the table definitions file in the Repl database by typing `prowin repl -1`.

   The **Procedure Editor** window appears.
6. Navigate to **Tools** > **Data Administration**.
7. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)…**.
8. In the **Load Data Definitions** window, load the `repl.df` file and click **OK**.

   Note: The `repl.df` file is located in the `db` folder of the Pro2 root folder.

9. Load table definitions in the Pro2 database by entering `prowin pro2 -1`.

   The **Procedure Editor** window appears.
10. Navigate to **Tools** > **Data Administration**.

11. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**.
12. In the **Load Data Definitions** window, load the `pro2.df` file and click **OK**.

    Note: The `pro2.df` file is located in *`Pro2 folder`*`/db`.

13. Start the repl and pro2 databases by typing:
    - `Proserve repl`
    - `Proserve pro2`
14. Add the Pro2 database connection details to the `replProc.pf` file in the Pro2 root folder under `\bprepl\Scripts\replProc.pf` to connect the replication threads and PAS for OpenEdge instance to the Pro2 database. Navigate to the `bprepl\scripts` folder and open the `replProc.pf` file. Add the complete path of the database in the file and click **Save**.
15. To check that both the Repl and Pro2 database instances are connected, open the **Procedure Editor** from the `bprepl\Scripts` folder and navigate to **Tools** > **Data Dictionary**. If the connection is successful, then the **Tables** section displays the tables.

Create an empty local copy of the source database

1. On your Pro2 machine, open a **Proenv** instance.
2. Use the `prodb` command to create an empty copy of your source database, for example:

   `prodb` *`source database name`* `empty`
3. Start the database copy by entering `proserve` *`source database name`*.
4. Add the source database connection details to the `replProc.pf` file in the Pro2 root folder under `\bprepl\Scripts\replProc.pf` to connect the replication threads and PAS for OpenEdge instance to the Pro2 database. Navigate to the `bprepl\scripts` folder and open the `replProc.pf` file. Add the complete path of the database in the file and click **Save**.
5. To check Repl and Pro2 database instances are connected successfully, open the **Procedure Editor** from the `bprepl\Scripts` folder and navigate to **Tools** > **Data Dictionary**. If the connection is successful, the **Tables** section displays the tables.

Start the PAS for OpenEdge instance and log in to the Pro2 web application

Now that the database instances are set up successfully, start the PAS OpenEdge instance and log in to the Pro2 web application.

1. Start a **Proenv** instance.
2. Navigate to the **Pro2** folder by entering `cd` followed by name of the folder where you installed Pro2, for example, `cd Pro2v62`.
3. Enter *`instance name`*`\bin\tcman.bat env`.

   Note: The default instance name is Pro2Web.

4. Type *`instance name`*`\bin\tcman.bat start`.

5. To verify that the instance started successfully, type *instance name*\bin\tcman.bat env. The server running section shows a number if the instance started.
6. Open any browser and log in at localhost:*port number*/pro2/static/ using the default credentials.

Load the configuration file

Before you set up the source and target database connections in the user interface, you must set up a basic configuration by loading the default configuration file. The configuration file consists of all the default properties and records that are necessary to set up the Pro2 environment.

Note: If you use Internet Explorer to load the configuration file, you may encounter errors. To avoid them, use an alternative web browser like Google Chrome.

To load the configuration file:

1. In the Pro2 web interface, navigate to **Actions** > **Tools** > **Load Configuration**.
2. Click **Select Files** and load the .ini configuration file, for example replbasev610.ini, from the Pro2 folder.
3. Click **Submit**. A success or failure notification is displayed after the upload is complete.

You can view the default properties and records on the **Properties** tab after the configuration file is successfully loaded.

Connect to your WAN application server instance

After the properties are loaded, you must connect to your WAN PAS for OpenEdge instance from the Pro2 user interface.

To connect to your WAN instance:

1. In the Pro2 user interface, click **Settings** from the side menu.
2. Select **LAN/WAN** from the options.
3. Choose **WAN**.
4. Enter your application server details and click **Submit** when complete. The Pro2\predefs.i file is updated.

Start the job runner

To start the job runner and initiate the scheduled Enterprise Push job, use the following procedure:

1. Navigate to the Pro2 Scripts folder, for example: C:\Pro2\bprepl\Scripts.
2. Select and execute the jobrunner.bat file.
3. Navigate back to the web interface and confirm that the jobrunner.bat is running by checking the status in the Pending Jobs watch-box.

Add the source database details

You can set up either trigger-based replication or CDC-based replication. In a trigger-based-replication, the replication queue records (also known as ReplQueue records) are generated by deploying Pro2 triggers on a source database. In a CDC-based replication the ReplQueue records are generated from change tracking table. However, for CDC-based replication, the source database must be CDC enabled.

To add source database details:

1. From the Pro2 web interface, navigate to the **Manage Replication** tab.
2. Click **New**.

   The **Create Replication** window appears.
3. (Optional) Enter your **User Name** and **Password**.

   Note: Enter your user name and password only if it is required by your source database.
4. Select **Test Connection**.

   If the test is successful, move on to set the target database. Review your source database details to ensure they are correct.
5. Click **Next**.

Generate and load the source schema file

By generating the source schema file, you build the files necessary to enable your local empty copy of your source database to communicate with the application server-side source database.

To generate and load the source schema file:

1. On the **Generate Source Schema** tab, click **Generate Source Schema**.
2. After the source schema is generated, click the **Download Schema files** and download all available files.
3. Copy and paste the source schema file *database name*_REM_AREAS.df in the db folder of Pro2.
4. Open a **Proenv** instance and enter the command `prowin` *database name* for the local empty copy of source database.
5. In the Procedure Editor, select **Tools** > **Data Administration**.
6. Select **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**
7. Add the source schema file as the **Input File** and click **OK**.
8. Navigate to the Pro2 web interface and click **Next**.

Add the target database details

The target database can be either a Microsoft SQL Server, Oracle database, or OpenEdge database. If your target database is an OpenEdge database, the you can forgo some of the configuration processes like generating the schema holder. However, there for a Microsoft SQL Server or Oracle database there are several configuration considerations to take into account. For more information about configuring Microsoft SQL Server or Oracle as a target database, see Configure Pro2 to target SQL database connectivity on page 69.

To add target database details:

1. Select **Target Database Type**. Choose Microsoft SQL Server, Oracle database, or OpenEdge databases.
2. Following fields are automatically filled, but can be changed if necessary
   - Target Database Name
   - Target Schema Image
   - Schema Holder DB
   - Target ODBC Connection
3. (Optional) Enter your user name and password.

Note: Entering your user name and password only if it is required by your target database.

4. Click **Next**.

   The **Generate Target Schema** tab appears.

Generate the target schema file

By generating the target schema file, you load the target database with files necessary to build out the target database. Pro2 reads the source database schema, and generates scripts that are then executed on the target database server to create the target schema. By default, all tables and fields are created in the target. The rest of the files (`drop.sql`, `index.sql`, `trunc.sql`, `errors.log`, and `warnings.log`) are created separately. You can execute these files as necessary. Typically, tables that are not replicated remain unmapped.

Pro2 reads the source Progress schema and generates scripts that are then executed on the target database server to create the target schema. By default, all tables and fields are created in the target. Typically, tables not replicated remain in the target bur are simply not mapped for replication, however, if it is desired that the tables do not exist at all on the target side, the generated scripts can be edited to remove the pertinent statements.

To generate target schema file:

1. Click **Generate Target Schema**. A table that contains a replication job appears.

   Note: Do not execute this job until you run the job runner file. A job runner batch file continuously processes replication jobs from the job queue. This file is in the `Scripts` folder.

2. Click **Download** under the **Download Schema files** column, to download the target schema files.

Create a target database

If you have not done so already, create a target database for you replicated data. The target database can be either a Microsoft SQL Server, Oracle, or OpenEdge database. Creating a target database will depend largely on they type of database you want to use and your development tooling. The following lists the general steps to create your target database:

1. In your database admin tool, like Microsoft SQL server Management Studio, create a new database and give it a name.
2. Create a new database query, and load the schema file into the query.

For more information about target databases, see

Create a schema holder for your target database

An empty OpenEdge database is created to act as the schema holder for the Microsoft SQL Server or an Oracle database. The schema holder is the layer that allows ABL to interact with the Microsoft SQL Server or Oracle database as though it were a native OpenEdge database.

To create a schema holder:

1. Open a **Proenv** window, and type `prodb` *`database name`* `empty` to create an empty schema holder database.
2. Start a single user session by typing `prowin` *`database name`* `-1`.

   The **Procedure Editor** window appears.

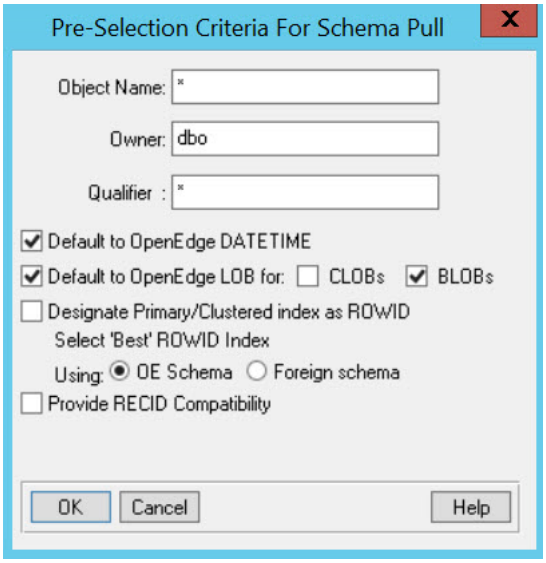
3. From the **Procedure Editor**, select **Tools** > **Data Administration**.

4. Select **DataServer** > ***database type*** **Utilities** > **Create DataServer Schema** .

   The **Create/Modify Record for Schema** window appears.

5. Specify the **Logical Database Name**.
6. If the source database is a SQL Server, specify the **ODBC Data Source Name**. Click **OK**.
7. Enter the login ID and password if required in the next dialog box and click **OK**.
8. Enter criteria for **Object Name**, **Object Owner** (typically, `dbo`), and **Qualifier**.
9. Select the **Default to OpenEdge DATETIME** option.
10. If you are using LOB fields in replication, ensure that the **Default to OpenEdge LOB** for **BLOBs** option is selected along with the CLOBs option. This forces the data server to automatically consider certain target side data types to be treated as LOBs.

    By selecting **CLOBs**, `varchar(max)/nvarchar(max)` data types are automatically converted to CLOBs. By selecting **BLOBs** the data server is forced to consider all `varbinary(max)` as BLOBs. On source fields that are character types, Pro2 may convert some types to `varchar(max)` when the source width is wider than the `MAX_CHAR_WIDTH` property setting. This conversion has the added benefit of avoiding row size limits for SQL Servers. However, `varchar(max)/nvarchar(max)` are not necessarily treated as CLOBs.



11. Click **OK** to finish.

Load the target database table definitions

After the target schema file is downloaded, load the target database with the appropriate tables required for replication.

For OpenEdge target databases:

1. Copy and paste the target schema file in the `db` folder of Pro2.
2. Open a Proenv window, enter `prowin` *`target database name`*.
3. In the Procedure Editor, select **Tools** > **Data Administration**.
4. Select **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**

5. Add the target schema fileas the **Input File** and click **OK**.
6. Navigate to the Pro2 web interface and click **Next**.

   The **Mapping Source to Target Tables** tab appears.

For Microsoft SQL Server or Oracle target databases:

1. Open a **Proenv** window, type `prowin` *`database name`* `-1` and press **Enter**.

   The **Procedure Editor** window appears.
2. Select **Tools** > **Data Administration**.
3. Select **Database** > **Select Working Database**.
4. Click **OK**. The **Data Administration** window appears.
5. Select **DataServer** > ***`database type`*** **Utilities** > **Update/Add Table Definitions…**

   The **Pre-Selection Criteria for Schema Pull** window appears.
6. Enter `dbo` for the **Owner**, select the **Default to OpenEdge DATETIME** and **Default to OpenEdge LOB for: CLOBs and BLOBs** checkboxes. Click **OK**.

   The **Select *`database type`* Objects** window appears.
7. Click **Select Some...**.

   The Select Objects by Pattern Match window appears.
8. Enter `dbo` for the **Owner**, `table` as the **Object Type**, and click **OK**.
9. Truncate the *`database name`* database to restrict operations in this database. This database should be a read-only database. Enter the following command in the **Proenv** window.

   `proutil` *`database name`* `-C truncate bi`
10. Navigate to the Pro2 web interface and click **Next**.

    The **Mapping Source to Target Tables** tab appears.

Map tables and fields

You must map data fields between your source and target database so that data in the source can be replicated to the target. You can do this from the **Mapping** dashboard in the web interface. The repl database contains all the mapping information for the replication of the databases, tables, and fields between the source and target. This information is stored in the `ReplDBXref, ReplTableXref,` and `ReplFieldXref` tables. Mapping information can be saved to and loaded from a text file.

Depending on the number of schemas, primary keys, and foreign keys of the databases, the data sources and the database mapping information has varying degrees of complexity.

Map source tables to target tables and generate code

To map source tables to target tables and generate code:

1. On the **Mapping Source to Target Tables** tab, select the tables from the source database that you want to map with the target database. You can map the tables in the following ways:
   - Map—Choose the **Map** option if you want to map a single source table with a single target table.
   - Automap—Choose the **Automap** option if you want to map one or more source tables to the corresponding target tables.

Note: You can also unmap the tables based on your needs using the **Unmap** option.

2. Select the tables to map, click **Automap**, and click **Next**.
3. In the **Advanced Configuration** tab, change thread assignments and other optional control flags if required.

Note: The **Advanced Configuration** tab is an optional settings tab.

4. Click **Next**.

   The **Generate Code** tab appears.

5. Click **Generate Code**. You are directed to the Pro2 web interface dashboard. Click the **Pending Jobs** watch-box to see which replication jobs are running.
6. Run the replication file to execute the replication jobs. In **Windows Explorer**, navigate to `\bprepl\Scripts`.
7. Double-click the **Replbatch1.bat** file.

   The corresponding replication thread starts running on the **Dashboard**.

Note: In a CDC-based replication, double-click the **CDCbatch.bat** file.

8. Refresh the Pro2 user interface to complete the replication configuration process.

Manually test a replication

Testing is an important part of any enterprise applications configuration process. Consider an testing example where a city is assigned to a customer. This change is made in the source database, and needs to be replicated to the target database. To perform the test scenario:

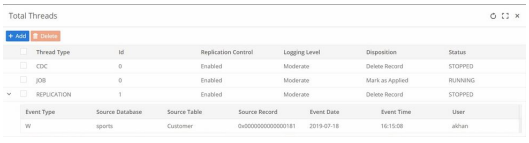
1. In a **Proenv** window, navigate to your Pro2 root folder. For example, `C:\Pro2v62\db`.
2. Enter the command `prowin -db` *`database name`* `-db repl`.

   The **Procedure Editor** window appears.

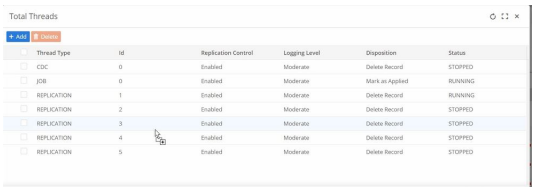
3. Assign city to the first customer in source database.

```
FIND FIRST customer
ASSIGN city = “NY”
```

4. Open the Pro2 web interface and click the **Total Threads** watch-box. Notice the event type in the **Replication thread** window. This event will show the recent change made to the source database.



5. Run the replication file by navigating to `\bprepl\Scripts` .
6. Double click the **Replbatch1.bat** file to run the replication thread. This action removes the event type in the replication thread which implies that the replication record is processed by the replication thread and sent to the target database.



Directory reference

The following is a description of the folders under the **Pro2** root folder.

Folder	Description
`bprepl`	Root directory for application programs
`bprepl\AppSrv`	Used in WAN implementations
`bprepl\images`	Image files that can be used for shortcut icons
`bprepl\misc`	Miscellaneous replication files and procedures
`bprepl\repl_as_tgt`	Used in WAN implementations
`bprepl\repl_d`	Directory for the Pro2 generated database replication delete trigger procedures
`bprepl\repl_export`	Used in WAN bulk loads
`bprepl\repl_inc`	Directory for Pro2 generated assigned include files
`bprepl\repl_jtrig`	Directory for any Java triggers
`bprepl\repl_log`	Default location of log files
`bprepl\repl_mgtrig`	Directory of tables that require merged triggers
`bprepl\repl_mproc`	Directory for the Pro2-generated bulk copy procedures
`bprepl\repl_proc`	Directory for the Pro2-generated replication library
`bprepl\repl_w`	Directory for Pro2-generated database replication write trigger procedures
`bprepl\repl_tmpl`	Directory containing the templates used for various code generation
`bprepl\Scripts`	Directory containing the `.pf` files, scripts, and shortcuts to start the Pro2 Admin Tool, replication processors, and bulk load procedures
`bprepl\SQL_inc`	Directory for the direct, SQL assigned include files
`bprepl\SQL_mproc`	Directory for the direct, SQL bulk copy procedures
`bprepl\SQL_proc`	Directory for the direct, SQL replication procedures

Folder	Description
custom	Directory to deploy your customized code so that it overrides the existing code in the bprepl folder **Note:** This folder must maintain the bprepl folder structure and is applicable for both LAN and WAN configurations
db	Location of the schema holder database. Also, the initial temporary location for Repl database during implementation
Installation	Initially empty. Used to save installation site-specific files
misc	Miscellaneous Pro2 utilities
tmp	Miscellaneous Pro2 temporary files used during implementation

ReplProperties and ReplQueue

The OpenEdge replication source database and target database tables are stored in property files. There are configuration settings whose values you set in these tables that control different aspects of your replication environment. The following are the tables for the Pro2 replication environment:

- ReplProperties—Configuration settings such as log file location, logical delete tables, and specification of procedure templates are stored in the ReplProperties table. Configuration settings can be saved to and loaded from a text file from the **New** button in the **Manage Replication** window on the Pro2 web interface.
- ReplQueue—Information about change events is stored in the ReplQueue table. This information includes the ROWID of the record changed, event date, time, and queue thread. Typically, repl queue records represent updates made to the source database that are waiting to be written to the target SQL database.

Install and configure Pro2 on Linux

Use the Pro2-6.1.bin file to install Pro2 and configure a standard instance of the Pro2 web interface, or an Enterprise View instance, on your Linux machine.

Install Pro2

To install Pro2 for Linux standard or Enterprise View:

1. Locate and execute either the **Pro2-6.1-ent.bin** file for Enterprise View, or the **Pro2-6.1.bin** file for standard.
2. In the **Pro2** installer, click **Next** to start the installation process.
3. Specify the install folder path, for example, `USER_INSTALL_DIR=/largedisk/<user>/pro2inst`.
4. In the **Choose OpenEdge** window, specify the path or select **Choose** to browse and set the OpenEdge home directory.

   The directory is based on the version of OpenEdge that is installed on your machine, for example, `OE_INSTALL_DIR=/largedisk/<user>/120/linuxx86_64/build/dlc`.
5. On the PAS for EpenEdge Ports screen, enter the port information for your PAS for OpenEdge instance and designate the instance name. For example:

HTTP Port	9991
HTTPS Port	9992
Shut Down Port	9993
Instance Name	Pro2Web

6. Click **Next** and review the options you chose for this installation in the **Pre-Installation Summary** page.
7. Click **Install**.
8. Click **Done**.

Create a Pro2 database

The repl and pro2 database instances contain the replication statistics data that is displayed on the **Dashboard** tab of the Pro2 web interface.

Use the following procedure to create a Pro2 database:

1. Start a Proenv session.
2. From the **Proenv** window go to the **Pro2 folder** > **db**

   Note: Check the directory of the **db** folder to view the database instances.

3. Create an empty `pro2ent` database instance by entering the `prodb pro2 empty`command.
4. Load the required table definitions in the `pro2ent` database by entering the `prowin pro2 -1`.

   The **Procedure Editor** window appears.
5. Navigate to **Tools** > **Data Administration**.
6. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**.
7. In the **Load Data Definitions** window, load the `pro2.df` file and click **OK**.

   Note: The `pro2.df` file is located in *`Pro2 folder`*`/db`.

8. Start the `pro2ent` database by typing: `Proserve pro2`

Update `replProc.pf`

In the `pro2` database you must add the connection details to the `replProc.pf` file. This connects the database to your PAS for OpenEdge instance, and by extension, makes it available to be connected to your other Pro2 instances.

1. Navigate to the **Pro2ent** folder.
2. Add the `pro2ent` database connection details to the `Pro2_HOME\bprepl_ent\Scripts\replProc.pf` file to connect the replication threads and PAS for OpenEdge instance to the `Pro2ent` database.
3. Navigate to the `bprepl_ent\scripts` folder and open the `replProc.pf` file. Add the complete path of the database in the file and click **Save**.
4. To test the `Pro2ent` database connections, open the **Procedure Editor** from the `bprepl_ent\Scripts` folder and navigate to **Tools** > **Data Dictionary**. If the connection is successful, the **Tables** section displays tables.

Start a PASOE instance

Now that the `pro2ent` database is set up, start the PAS for OpenEdge instance and log in to the Pro2 web application.

1. Start a Proenv session.
2. Navigate to the Pro2 folder where you installed Pro2
3. Type `Pro2LAN\bin\tcman.sh env`
4. Type`Pro2LAN\bin\tcman.sh start`
5. To check if the instance has started successfully, type `Pro2LAN\bin\tcman.sh env`. The server running section shows a port number if the instance has started.
6. Open any browser and log in at `localhost:9991/pro2/static/` with the default credentials.

Setup the Pro2 environment

Before using the Pro2 web interface, you must set up basic configuration by loading the default configuration file. The configuration file consists of all the default properties and records which are necessary to set up the Pro2 environment.

To load the configuration file:

1. In the Pro2 web interface, navigate to **Settings** > **Tools** > **Load Configuration**.
2. Click `Select Files` and load the `replentbasev610.ini` file from the Pro2 folder. The `replentbasev610.ini` file is Pro2 the configuration file.
3. Click **Submit**.

If you have been using the **Pro2-6.1-ent.bin** file to set up Enterprise View on your Linux machine, use the following procedures to finish the Enterprise View configuration process.

Join your Pro2 instances to the Pro2 Enterprise View

After the configuration file is uploaded, your instance of Pro2 Enterprise View is ready to be joined with your other instances of Pro2. In this phase of the set up process, you transform Enterprise view from an empty dashboard, to one full of meaningful data from across all of your Pro2 instances.

Note: A Pro2 instance cannot be joined to more than one Enterprise View instance.

To join your Pro2 instances:

1. From an instance of Pro2 that is not joined with Enterprise View, navigate to the side menu, and select **Settings** > **Enterprise** > **Join**.
2. On the **Join Enterprise** screen, enter:
   a. **Instance name**

      The instance name is designated at the time of installation and setup of that instance.

   b. **Instance Abbreviation**

      The instance abbreviation would also be designated at the time of installation and setup of that instance. The abbreviation is limited to two characters.

   c. **Enterprise Application Server Name**

      This is the same application server that is associated with your Enterprise View instance, for example, `ent`.

   d. **Enterprise Host Name**

      This is the same host name that is associated with your Enterprise View instance, for example, `localhost`.

   e. **Enterprise Application Server Port Number**

      The port number associated with your enterprise application server.

3. (Optional) Test your connection to the application server with the **Application Server Test Connection** button. If the test is unsuccessful, ensure that the instance information is correct and try again.
4. Select **Join** to connect the Pro2 and Enterprise View instances together. Alternately, you can save your work and come back to it later by using the **Save** button.

Note: It takes a few minutes for the instances to connect.

Schedule an Enterprise push job

After the connection between your Enterprise View instance and your other Pro2 instances are established, data is not automatically transferred between them. You must schedule an Enterprise push job. Your Pro2 instance to push data, metrics, and reports from an instance to the Enterprise View instance. This is how the Enterprise View instance stays up to date with all of your mission-critical replications.

To Schedule an Enterprise Push job:

1. From an instance of Pro2 that is joined with Enterprise View, navigate to the side menu, and select **Actions** > **Jobs** > **Scheduled Jobs**.
2. On the **Scheduled jobs** window, select **New**.
3. On the **Create Scheduled Jobs** window, click on the **Type** drop-down list and select **Enterprise Push**.
4. (Optional) Enter a description for the job. Entering a description can help you differentiate various jobs of the same type from one another.
5. Under the **Settings** section, set how frequently the data is to be pushed to your Enterprise View instance. It is a best practice to schedule the pushes to occur frequently. To support this:
   a. Select **Daily**.
   b. Enter the **Start Date** and **Start Time** for when you want the job to begin, such as the current date and time.
   c. You can set the **End Date** and **End Time** for the job. This, however, stops the flow of data from your Pro2 instance to your Enterprise View instance and is not recommended. To run the job indefinitely, **do not enter** an **End Date** or an **End Time**.
   d. Set the **Repeat task every** to **1 minute**.
6. Click **Save**.

Start the job runner

For the last step of the Enterprise View setup process, you must start the job runner.

To start the job runner and initiate the scheduled Enterprise Push job, use the following procedure:

1. On the machine where you have the Pro2 instance for which you have scheduled the job, navigate to the Pro2 Scripts folder, for example `C:\Pro2\bprepl\Scripts`.
2. Select and execute the **jobrunner.sh** file.
3. Navigate back to the Pro2 instances web interface and confirm that the job completed successfully.

Your Enterprise View instance is now set up, and data will begin to populate the dashboard.

Pro2 Enterprise View

The Pro2 Enterprise View is a centralized dashboard that aggregates mission-critical data, reports, and metrics across all of your Pro2 instances. From the Enterprise View dashboard, you can quickly access each of your Pro2 instances to manage and act on any reports or replications from the instance.

For example, an e-commerce enterprise has three instances of OpenEdge Pro2 running on three separate data centers: one in the USA, one in India, and one in the UK. Managing data and replications across these three instances can be a hassle. With the Enterprise View, you can observe and navigate to each Pro2 instance with ease and efficiency.

The Pro2 Enterprise View dashboard is included with the latest release of Pro2, but it can also be installed as an add-on to your existing instances. If you are interested in the Pro2 Enterprise View Dashboard, contact your Progress OpenEdge representative, and ask about upgrading to the latest version, or about adding the Pro2 Enterprise View Dashboard to your existing instances.

Main steps to set up Pro2 Enterprise View

The Pro2 Enterprise View is designed for easy installation, set up, and integration with your existing Pro2 instances.

To set up Pro2 Enterprise View:

1. Install Enterprise View.
2. Create a Pro2 Enterprise View database.
3. Update the `replProc.pf` file.
4. Start a PAS for OpenEdge instance.
5. Set up your Pro2 Enterprise environment.
6. Join your Pro2 instances to Enterprise View
7. Schedule an Enterprise View push job.
8. Start the job runner.

Install Enterprise View

To install Enterprise View:

1. Locate and execute the `Pro2-6.1-ent.exe` file. All installation files are provided to you by your Progress representative.
2. In the **Pro2** installation wizard, click **Next**.
3. Specify the installation folder path as `C:\Progress\Pro2ent`.
4. In the **Choose OpenEdge** window, specify the path, or select **Choose** to browse and set the OpenEdge home directory.

   The directory is based on the version of OpenEdge that is installed on your machine, for example, `C:\120\`.
5. On the **PAS Ports** window, enter the port information for your PAS for OpenEdge instance, and designate the instance name.
6. Click **Next** and review the options you chose for this installation in the **Pre-Installation Summary** page.
7. Click **Install** to continue with the installation process.
8. Click **Done**.

Create a Pro2 Enterprise View database

Similar to a standard Pro2 replication (repl) database, the Pro2 Enterprise View, or `pro2ent`, database is an internal database that exists on the machine where Enterprise View is installed. All reports and metrics from the Pro2 instances that you joined to Enterprise View are aggregated to this database, where it is then pushed to the Pro2 web user interface.

To Create a pro2ent database:

1. Start a Proenv session.
2. Navigate to the `Pro2` folder from the Proenv window. Type `cd` followed by name of the folder where you installed Pro2 Enterprise View, for example, `cd Pro2ent`.
3. Type `cd db` to navigate to the database folder of Pro2.

Note: Check the directory of the `db` folder to view the database instances. Type `dir` after navigating to the database folder.

---

4. Create an empty `pro2ent` database instance by typing `prodb pro2ent empty`.
5. Load the required table definitions in the `pro2ent` database by typing `prowin pro2ent -1`.

   The **Procedure Editor** window appears.
6. Navigate to **Tools** > **Data Administration**.
7. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**.
8. In the **Load Data Definitions** window, load the `pro2ent.df` file and click **OK**.

   Note: The `pro2ent.df` file is located in *`Pro2ent folder`*`/db`.
9. Start the `pro2ent` database by typing: `Proserve pro2ent`

Update replProc.pf

In the `pro2ent` database, you must add the connection details to the `replProc.pf` file. This connects the database to your PAS for OpenEdge instance, and by extension, makes it available to be connected to your other Pro2 instances.

1. Navigate to the `Pro2ent` folder.
2. Add the `pro2ent` database connection details to the `Pro2_HOME\bprepl_ent\Scripts\replProc.pf` file to connect the replication threads and PAS for OpenEdge instance to the `Pro2ent` database.
3. Navigate to the `bprepl_ent\Scripts` folder, and open the `replProc.pf` file. Add the complete path of the database in the file and click **Save**.
4. To test the `Pro2ent` database connections, open the **Procedure Editor** from the `bprepl_ent\Scripts` folder and navigate to **Tools** > **Data Dictionary**. If the connection is successful, the **Tables** section displays tables.

Start a PAS for OpenEdge instance

Now that the `pro2ent` database is set up, start the PAS for OpenEdge instance and log in to the Pro2 web application:

1. Start a Proenv session.
2. Navigate to the Pro2 folder by entering `cd` followed by name of the folder where you installed Pro2, for example, `cd Pro2ent`.
3. Type `Pro2ent\bin\tcman.bat env`.
4. Type `Pro2ent\bin\tcman.bat start`.
5. To verify that the instance started successfully, type `Pro2ent\bin\tcman.bat env`. The **Server Running** section shows a number if the instance started.
6. Open any browser and log in at `localhost:9991/pro2/static/` with he default credentials.

Set up the Pro2 Enterprise environment

Before you connect your Pro2 instances with your Enterprise View instance, you must set up a basic environment by loading the configuration file. The configuration file consists of all the properties and records that are necessary to set up the Pro2 Enterprise environment.

To load the configuration file:

1. In the Pro2 web interface, navigate to **Settings** > **Tools** > **Load Configuration**.
2. Click **Select Files** and load the `replentbasev610.ini` file from the Pro2 folder. The `replentbasev610.ini` file is the Pro2 configuration file.
3. Click **Submit**.

Join your Pro2 instances to Pro2 Enterprise View

After the configuration file is successfully uploaded, your instance of the Pro2 Enterprise View is ready to be joined with your other instances of Pro2. In this phase of the set-up process, you transform Enterprise View from an empty dashboard to one that is full of meaningful data from across all of your Pro2 instances.

Note: A Pro2 instance cannot be joined to more than one Enterprise View instance.

To join your Pro2 instances:

1. From an instance of Pro2 that is not joined with Enterprise View, navigate to the side menu and select **Settings** > **Enterprise** > **Join**.
2. On the **Join Enterprise** window, enter:
   a. **Instance name**

      The instance name is designated at the time of installation and setup of that instance.

   b. **Instance Abbreviation**

      The instance abbreviation is determined at the time of installation and setup of that instance. The abbreviation is limited to two characters.

   c. **Enterprise Application Server Name**

      This is the same application server that is associated with your Enterprise View instance, for example, `ent`.

   d. **Enterprise Host Name**

      This is the same host name that is associated with your Enterprise View instance, for example, `localhost`.

   e. **Enterprise Application Server Port Number**

      The port number associated with your enterprise application server.

3. (Optional) To test your connection to the application server, click the **Application Server Test Connection** button. If the test is unsuccessful, ensure that the instance information is correct and try again.
4. Select **Join** to connect the Pro2 and Enterprise View instances. You can save your work and come back to it later by clicking the **Save** button.

Note: It takes a few minutes for the instances to connect.

Schedule an Enterprise Push job

After the connection between your Enterprise View instance and your other Pro2 instances are established, data is not automatically transferred between them. You must schedule an Enterprise push job. Your Pro2 instance to push data, metrics, and reports from an instance to the Enterprise View instance. This is how the Enterprise View instance stays up to date with all of your mission-critical replications.

To Schedule an Enterprise Push job:

1. From an instance of Pro2 that is joined with Enterprise View, navigate to the side menu, and select **Actions** > **Jobs** > **Scheduled Jobs**.
2. On the **Scheduled jobs** window, select **New**.
3. On the **Create Scheduled Jobs** window, click on the **Type** drop-down list and select **Enterprise Push**.
4. (Optional) Enter a description for the job. Entering a description can help you differentiate various jobs of the same type from one another.
5. Under the **Settings** section, set how frequently the data is to be pushed to your Enterprise View instance. It is a best practice to schedule the pushes to occur frequently. To support this:
   a. Select **Daily**.
   b. Enter the **Start Date** and **Start Time** for when you want the job to begin, such as the current date and time.
   c. You can set the **End Date** and **End Time** for the job. This, however, stops the flow of data from your Pro2 instance to your Enterprise View instance and is not recommended. To run the job indefinitely, **do not enter** an **End Date** or an **End Time**.
   d. Set the **Repeat task every** to **1 minute**.
6. Click **Save**.

Start the job runner

For the last step of the Enterprise View setup process, you must start the job runner.

To start the job runner and initiate the scheduled Enterprise Push job, use the following procedure:

1. On the machine where you have the Pro2 instance for which you have scheduled the job, navigate to the Pro2 Scripts folder, for example `C:\Pro2\bprepl\Scripts`.
2. Select and execute the **jobrunner.bat** file.
3. Navigate back to the Pro2 instances web interface and confirm that the job completed successfully.

Your Enterprise View instance is now set up, and data will begin to populate the dashboard. Repeat phases 6, 7, and 8 for each of your Pro2 instances that you want to join to your Enterprise View dashboard.

II

Manage

For details, see the following topics:

- Customize Pro2
- Upgrade Pro2 5.5 to 6.2
- Uninstall Pro2
- Use replication triggers
- Remove replication triggers
- Customize Pro2 replication
- Split replication threads
- Configure Pro2 to target SQL database connectivity
- Configure Pro2 e-mail notifications
- Use second-pass replication
- Bulk load with Pro2
- Dump and load with Pro2
- Migrate your OpenEdge database with Pro2
- Check replication logs
- How to schedule jobs in Pro2
- Use CDC with Pro2

- Manage Pro2 Properties
- GET bulk load report API
- GET replication queue records API
- GET thread data API
- GET Pro2 connection information API
- GET version information API
- GET instance information API

Customize Pro2

Pro2 can be customized to meet your needs. You can generate Pro2 specific fields, add and delete index information on the target database, and change table and field names.

Customizable by table, by field/column, or record/row

All tables or a subset of tables can be replicated. In addition, not all fields within the tables selected for replication need to be replicated. In addition, logic can be added to filter records in selected tables so that only those records that meet the specified criteria are replicated.

Custom transformations using ABL-supported logic

Replication data can be manipulated with ABL logic before being written to the target database. For example, field values can be modified based on business logic or target-side-only columns can be populated with data consolidated from multiple source fields.

Generating Pro2-specific fields

Pro2 provides an option Delta DF Pro2Pro to generate only Pro2-specific fields for your target database. In cases where the source schema is already replicated to your target database, you can use Delta DF Pro2Pro to generate only Pro2-specific fields, and thereby avoid unnecessary replication of source schema.

Oracle specialization: Use the following properties to generate Oracle-specific output:

- **`ORACLE_USE_LOGICAL`**—This property is responsible for converting the logical field to number.
- **`ORACLE_USE_SCALE`**—This property is responsible for adding precision and scale for decimal fields.

By default, the value of these properties is set to **NO**. To use these properties, set the value to **YES**.

Target-side index information

Customers can add and delete index information on the target database without interfering with Pro2 Replication as long as no unique keys are added and as long as the `PRROWID` key is not modified. Pro2 compiles the CREATE INDEX statements into a separate file named *`dbname -targettype`* `-index.sql` file. These index statements are applied to the target only if you designate them.

Standard modifications to table and field names

You can modify table and field names in Pro2.

- **Extent fields**—Array fields from the OpenEdge source database are created as individual fields in the target database schema.
- **Reserved words**—Any table or column names that are reserved words in the target database server or in OpenEdge must be renamed. Often this is done by appending an underscore (_) to the end of the column name. However, the column can be renamed to anything that is not a reserved word.

Upgrade Pro2 5.5 to 6.2

Upgrading Pro2 from 5.5 to 6.2 is a manual process that involves installing and configuring Pro2 6.2, importing your 5.5 properties file and table mapping, and disconnecting and shutting down your Pro2 5.5 instance.

Pro2 6.2 includes all of the enhancements and benefits of the previous 6.x releases and offers many advantages over the 5.x series. Namely, a new intuitively designed user interface, enhancements for change data capture based replication, efficient bulk loading, and so much more. Because of these drastic differences between the two version series, upgrading Pro2 can best be achieved by installing a fresh version of Pro2 6.2, preparing your 5.5 instance of Pro2 for shutdown, and initiating a cut over event to switch from using 5.5 to 6.2.

An aspect of the upgrade process that requires special consideration, is determining whether or not you want to bring your 5.5 properties file into your new instance of Pro2. Generally, older Pro2 installations were performed by Progress Professional Services as a paid engagement. As such, Pro2 configurations can vary greatly between one another depending on how thoroughly the default properties file and code was customized. During your upgrade planning phase, examine your properties an compare them to the new 6.2 properties.

It is highly recommended that you first attempt the upgrade on a test environment, and backup your entire Pro2 directory, before upgrading your production environment.

Main steps to upgrade Pro2

1. Plan for your upgrade
2. Save your 5.5 configuration files
3. Install Pro2 6.2
4. Configure Pro2 for LAN or WAN
5. Load you 5.5 configuration files
6. Initiate the upgrade cut-over event
7. Test your Pro2 6.2 deployment

Upgrade planning

Due to the highly customizable nature of Pro2 instances, there is no definitive upgrade planning method. Your upgrade planning needs might be vastly different than another Pro2 user. Confer with your database administrator as well as other stakeholders to determine how best to plan for your upgrade.

However, while planning for you upgrade there are a few guidelines to consider:

- Determine when your application receives the least amount of user traffic.
- Plan to initiate your upgrade cut over during a low traffic window.
- Learn about Pro2 6.2 features, functionality, and enhancements.
- Compare your custom 5.5 properties with the properties from 6.2. If your custom properties are still viable, create of copy of the file for later use.

  For more information about the Pro2 6.2 properties, see the Properties reference.
- Determine whether a LAN or a WAN configuration is right for your deployment. This can be a good time consider if your application needs a new configuration or if you want to stay with the same configuration. For example, changing from LAN to WAN.
- Identify where you want to install Pro2 6.2.
- Devise a test plan.
- Create a backup strategy.

Save your configuration files

If you determine that you want to bring your Pro2 5.5 configuration details into your 6.2 installation, use the following procedure to save your properties and mapping file. You can then import them into your Pro2 6.2 installation.

1. On the machine where you have Pro2 installed, open the Pro2 Administration tool and click **File** > **Save Configuration**
2. Save your configuration file to an easily accessible location.
3. Return to the Administration tool, click **File** > **Save mapping file**.
4. Save your mapping file to an easily accessible location.
5. In the Pro2 root folder, locate your `properties.ini` file.
6. Using a text editor, open the file to edit it.
7. Locate the `[Properties]` section in the file, and remove all of the other sections.
8. Go to the Pro2 root folder, and find the `mapping_TBLxRef.map` file
9. Open the file in a text editor, and add the following to reference to the section header `[Repl_TableXref]`, and save the file.
10. Next go to the `mapping_FLDxRef.map` file and add the following reference to the section header `[Repl_FieldXref`, and save the file.

    Save all the files that you modified in this section to an easily accessible location. You will need them during the Pro2 6.2 configuration process.

Install Pro2 6.2

If you choose to use the same configuration as your previous Pro2 deployment (LAN or WAN), then it is recommended to install Pro2 6.2 on the same machine as your 5.5 deployment. Keep in mind that, though this installation is on the same machine, it must be in a separate directory to ensure that there is no interference with your 5.5 test deployment of Pro2. By having both the 5.5 and 6.2 installation on the same machine, there are less configurations needed to support the cut-over event.

1. Depending on your choice of configuration, choose to install and configure 6.2 for LAN or WAN.
   - Install and configure Pro2 for LAN on page 15
   - Install and configure Pro2 for WAN on page 29
2. Follow the procedure as normal until you come to the step for loading your configuration file. Then use this procedure:
   a. In the Pro2 web interface, navigate to **Actions** > **Tools** > **Load Configuration**.
   b. Click **Select Files** and load the `.ini` configuration file, for example `replbasev610.ini`, from the Pro2 folder.
   c. Click **Submit**. A success or failure notification is displayed after the upload is complete.

      Ensure that all the properties were successfully loaded in to Pro2 database.
   d. On the Pro2 dashboard menu, click **Actions** > **Tools** > **Load Configuration**.
   e. Click **Select Files**
   f. Select the **Overwrite Pro2 Properties** check box.
   g. Load your Pro2 5.5 `.ini` configuration file, and click **Submit**.

      Ensure that all existing property values were overwritten with the Pro2 5.5 property values.
3. Start the job runner.
   a. Navigate to the Pro2 Scripts folder, for example `C:\Pro2\bprepl\Scripts`.
   b. Select and execute the `jobrunner.bat` file.
   c. Navigate back to the web interface and confirm that the `jobrunner.bat` is running by checking the status in the Pending Jobs watch-box.
4. Add the source database details
   a. From the Pro2 web interface, navigate to the **Manage Replication** tab.
   b. Click **New**.

      The **Create Replication** window appears.
   c. Enter **Source DB Name**
   d. Choose the **Source Database Mode**, either trigger or CDC.
   e. Enter **Host Name**
   f. Enter the port number in the `Host Port/Service` field (for example, 2233).
   g. (Optional) Enter your **User Name** and **Password**.

      Note: Enter your user name and password only if it is required by your source database.

   h. Select **Test Connection**.

      If the test is successful, move on to set the target database. Review your source database details to ensure they are correct.
   i. Click **Next**.

      The **Set Target** tab appears.

5. Add the target database details

   a. Select **Target Database Type**. Choose Microsoft SQL Server, Oracle database, or OpenEdge databases.

   b. Following fields are automatically filled, but can be changed if necessary

      - Target Database Name
      - Target Schema Image
      - Schema Holder DB
      - Target ODBC Connection

   c. (Optional) Enter your **User Name** and **Password**.

      Note: Enter user name and password only if it is required by your target database.

   d. Click **Next**.

      The **Generate Target Schema** tab appears.

6. Click **Generate Target Schema**.

   Note: If your target SQL database already exists and is no different than in your Pro2 v5.5 instance, then you do not need to generate the target schema or load the file into target.

7. Create a schema holder for your target database.

Note: There is no need to recreate the schema holder unless you are upgrading OpenEdge version on the Pro2 instance machine.

---

a. Open a **Proenv** window, and type `prodb` *`database name`* `empty` to create an empty schema holder database.

b. Start a single user session by typing `prowin` *`database name`* `-1`.

The **Procedure Editor** window appears.

c. From the **Procedure Editor**, select **Tools** > **Data Administration**.

d. Select **DataServer** > ***`database type`*** **Utilities** > **Create DataServer Schema** .

The **Create/Modify Record for Schema** window appears.

e. Specify **Logical Database Name**.

f. If the source database is a SQL server, specify the **ODBC Data Source Name**. Click **Ok**.

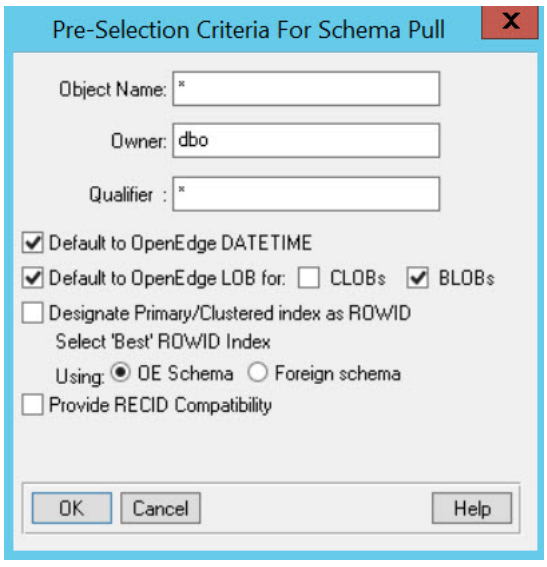
g. Enter the **Login ID** and **Password** if required in the next dialog box and click **OK**.

h. Enter criteria for **Object Name**, **Object Owner** (typically, `dbo`), and **Qualifier**.

i. Select the **Default to OpenEdge DATETIME** option.

j. If you are using LOB fields in replication, ensure that the **Default to OpenEdge LOB** for **BLOBs** option is selected along with the CLOBs option. This forces the data server to automatically consider certain target side data types to be treated as LOBs.

By selecting CLOB, `varchar(max)/nvarchar(max)` data types are automatically converted to CLOB. By selecting BLOB the data server is forced to consider all `varbinary(max)` as BLOB. On source fields that are character types, Pro2 may convert some types to `varchar(max)` when the source width is wider than the `MAX_CHAR_WIDTH` property setting. This conversion has the added benefit of avoiding row size limits for SQL Servers. However, `varchar(max)/nvarchar(max)` are not necessarily treated as CLOBS.



k. Stay on the Generate Target Schema page and **open a new tab**.

l. On the new tab, navigate to**Actions** > **Tools** > **Load Map File**

Note: The load map functionality will delete all the existing repl_tablexref and repl_fieldxref records and recreate

m. Load the v5.5 `mapping_TBLxRef.map` file.

n. Load the v5.5 `mapping_FLDxRef.map` file.

o. Return to the previous tab (Generate Target Schema page).

p. Click **Next** to navigate to Mapping Page.

8. Review the tables and fields mapping info from your v5.5 instance and click **Next**.

9. On the Advanced Configuration page, map your tables to threads. This is similar to version 5.5 mapping.

10. Click **Next** to generate the code.

Note: To ensure proper database connectivity, it is highly recommended to restart both jobrunner and PASOE WEB instance.

Initiate the upgrade cut-over event

Leaving the target database details unconfigured in the Pro2 UI, go to the Pro2 5.5 root folder and start a **Proenv** instance to shut down the 5.5 replication databases.

1. Open a **Proenv** window and use the `cd` command to navigate to the Pro2 5.5 root folder.

2. Type `proshut repl -by` to shut down the replication (repl) database.

3. Type `proshut replperf -by` to shut down the replperf database.

4. Return to the Pro2 6.2 user interface and continue the target side configuration process.

Test your Pro2 6.2 deployment

After confirming that the bulk load was successful, use the test plan that you created during your planning phase to test and troubleshoot the Pro2 6.2 deployment. Review the results of your testing, and note of any abnormalities. Investigate ways to correct any abnormalities you find, and make the appropriate the corrections.

When there are no abnormalities in your testing environment, choose a time to upgrade your production environment.

Uninstall Pro2

To uninstall Pro2 on Windows:

1. Delete the PAS for OpenEdge instance.

a. Open a **Proenv** window and type `pasman instances`.

b. Enter `pasman delete` *`instance name`* to delete the PAS for OpenEdge instance. Type YES to confirm.

c. Close the **Proenv** window.

2. Delete the database instance.

a. Open a **Proenv** window and navigate to the `Pro2` folder by entering `cd` followed by the name of the Pro2 folder, for example, `cd Pro2_61`.

b. Enter `cd dir` to navigate to the Pro2 directory.

c. Enter `proshut repl -by` to shut down the repl database.

d. Enter `proshut` *`source database name`* `-by` to shut down the source database.

3. Enter `cd..` to get to the parent folder.

4. Enter `del db` and then type Y to delete the database instances.

5. Go to the Pro2 folder and delete the folder skeleton.

Use replication triggers

Pro2 replication schema triggers are used with the source OpenEdge Database. A schema trigger is a `.p` procedure that you add, through the data dictionary, to the schema of a database. Schema trigger code defined in the OpenEdge database is executed by database clients.

Pro2 replication triggers are used with the source OpenEdge database. You can use these triggers to:

- Create the replication record based on the database, table, and row ID.
- Record the type of event, (create, write, or delete) that takes place on a record.
- Set the `Applied` flag to false (default).
- Define what happens when an event occurs on a record.

When you use an OpenEdge database architecture, the Pro2 replication triggers are executed by the client executable, as opposed to the trigger being executed by the database server executable, as in most other database systems.

The Pro2 trigger procedures are specified by the schema trigger definition. The Pro2 trigger procedures are lightweight and do not alter the your application flow in any way. They are transparent to both the application and the end-users.

In cases where replication triggers already exist in the source database for particular events and tables, the Pro2 replication logic needs to be integrated with or merged into the pre-existing triggers.

All replication triggers are compression triggers by default.

Note: You cannot use the compression trigger template when Auditing is implemented. You must override the default triggers.

Schema trigger event definitions

Schema triggers are fired when schema-level objects (tables) are modified and when the user log-on or log-off events occur. Some of these triggers are:

- CREATE: Creates and enables a database trigger.
- DELETE: Removes a database trigger.
- WRITE: Writes and edits a database trigger.
- RE-DEL (REPLICATION-DELETE): Removes a replication trigger.
- RP-WRI (REPLICATION-WRITE): Creates or updates a replication trigger. The specified procedure is run when this type of event occurs.

To get a trigger definition report from the Procedure Editor (The Procedure Editor is an OpenEdge ABL code editor):

1. Run **Tools** > **Data Administration**.
2. From the **Data Administration** window, select **Database** > **Reports** > **Triggers**.

The following table is an example of the output from the Progress trigger report. This example is of a `Customer` table in the `sports` database after Pro2 triggers were inserted:

Table 1: Progress trigger report

Table	Event	CRC	Flags	Procedure
CUSTOMER	CREATE	No		C:\dlc101c\sports\crcust.p
	DELETE	No		C:\dlc101c\sports\delcust.p
	RP-DEL	No		.\bprepl\repl_d\dsports_Customer.t
	RP-WRI	No		.\bprepl\repl_w\wsports_Customer.t
	WRITE	No		C:\dlc101c\sports\wrcust.p

Alternatively, these triggers can be displayed by the `replTrigDisp.p` procedure (to display) and the `replTrigDisp2.p` procedure (to file).

Trigger procedures

The Pro2 trigger procedures are found by the client executable in a location specified in the `PROPATH`. The replication trigger location can be in the same directory as the other triggers (if there are other triggers defined for the database). However, these triggers must be already included in the `PROPATH`. You must place the trigger procedures on the database server and on any application servers that end users run your application.

Note that the pathname specified for the Pro2 replication triggers in the schema trigger definition is a relative path starting with `bprepl`. Typically, a `bprepl` directory is created with the following sub-directories in a location already in the `PROPATH`:

- `repl_d` (delete triggers)
- `repl_w` (write triggers)

`PROPATH` can be modified to include the location of the `bprepl` directory.

When a replication trigger is defined for a table schema, the client attempts to run the specified trigger procedure. This happens when a client executable attempts to update a record in the source database.

The trigger procedures should be located on the database server if the user is connected as a local direct connect client (whether an end user or by PAS for OpenEdge servicing a remote client). Whereas, the trigger procedures should be on the client system if they are remote and are connected to the database via client/server.

For optimal performance, the replication triggers should be compiled with `.r code`. To compile the triggers, the standard OpenEdge application compiler is used in a Procedure Editor session connected to both the source database and the repl database.

Merged triggers

The Pro2 replication triggers must be merged with the original triggers if the pre-existing replication triggers (`REPLICATION-DELETE` or `REPLICATION-WRITE`) exist in the source database. You can merge the triggers by adding a RUN statement in the original replication trigger procedure that calls the Pro2 trigger or by adding the Pro2 replication trigger logic to the original trigger.

Note:

The types of triggers in Progress are as follows:

- **Primary triggers**—`CREATE`, `WRITE`, and `DELETE` are primary triggers, and they should not be integrated with Pro2 triggers. They are not affected by Pro2 replication.
- **Replication triggers**— `REPLICATION-DELETE` or `REPLICATION-WRITE` are executed after the primary triggers and must be merged with the Pro2 triggers.

Insert replication triggers

The `replTrigInsert.p` procedure is used to insert the triggers. This procedure is run from the OpenEdge procedure editor connected to both the source database and the repl database (or to the source database with the embedded repl tables).

To insert replication triggers:

1. Shut down all the database instances and PAS for OpenEdge instance before inserting the replication triggers.
2. Navigate to the `\db` folder of Pro2 from the **Windows Explorer**, for example, `C:\Pro2v61\db`).
3. Create a new folder named `bprepl` in the `db` folder.
4. Navigate to the `\bprepl` folder in the Pro2 folder (`C:\Pro2v61\bprepl`), and copy the following folders:
   - `repl_d`
   - `repl_w`

   These folders contain the replication triggers and are required to process the replication queue records (also known as replq records).
5. Paste the folders in the `\db\bprepl` folder of Pro2 (`C:\Pro2v61\db\bprepl`).
6. Type `prowin -db sports -db repl` and press **Enter**.

   The **Procedure Editor** window appears.
7. Open the `ReplTrigInsert.p` file from `C:\Pro2v61\utils` in the **Procedure Editor** window. This inserts replication triggers into the source database.
8. Enter the name of the source database and close the **Procedure Editor** .
9. Restart all the database instances and the PAS for OpenEdge instance.

Remove replication triggers

The `replTrigDel.p` procedure is used to delete the replication triggers. This procedure is run from the OpenEdge procedure editor connected to both the source database and the replication database (or to the source database with the embedded replication tables).

To delete replication triggers:

1. Shut down all the database instances and PAS for OpenEdge instances prior to deleting the replication triggers.
2. Navigate to the `\db` folder of Pro2 from the **Windows Explorer**, for example, `C:\Pro2v61\db`).
3. Create a new folder named `bprepl` in the `db` folder.
4. Navigate to the \bprepl folder in the Pro2 folder (`C:\Pro2v61\bprepl`) and copy the following folders:
   - `repl_d`
   - `repl_w`

   These folders contain the replication triggers and are required to process the replication queue records (also known as replq records).
5. Paste the folders in the `\db\bprepl` folder of Pro2 (`C:\Pro2v61\db\bprepl`).

   Note: Ignore steps 4 and 5 if the triggers are already added in the `\db\bprepl` folder.

6. Type `prowin -db sports -db repl` and press **Enter**.

   The **Procedure Editor** window appears.
7. Open the `ReplTrigDel.p` file from `C:\Pro2v61\utils` in the **Procedure Editor** window.
8. Enter the name of the source database (for example, `sports`), and close the **Procedure Editor** window.
9. Restart all the database instances and the PAS for OpenEdge instance.

Note: Running the `replTrigDel.p` does not delete the trigger procedures. After the trigger definitions are removed, the trigger procedures do not execute even if they remain in the system.

Customize Pro2 replication

Pro2 is written in ABL and is designed for scalability and customization. You can customize auditing, source changes capturing, record maintenance, datatype, and name transformation.

Capture source changes by SQL clients

The standard Pro2 replication triggers capture changes made by ABL clients. Pro2 can also capture changes made to the source OpenEdge database by SQL clients. To do this, Java triggers can be implemented in the OpenEdge source database.

Audit database

Pro2 can be extended to write to a second target database to maintain audit records. Whereas each row is unique in the primary replication database (using PRROWID column), the audit database has multiple rows corresponding to each source database record, with a new row created for each change made to the source record. Auditing is implemented by having a second pass through the replication queue by a separate replication processor designated for the audit database. Typically, when auditing is implemented, a subset of the replicated tables is audited, however; it is possible to audit all tables.

Logical deletes

There are times when auditing is not required, but you want to maintain records in the target database after they are deleted from the source database. In this case, instead of being deleted, rows are designated as logical deletes (using PRROWID modification) and the data is kept in the target database.

Data-type and name transformation

Data-types on the target side can be the same as on the source database, or they can be different. For example, precision of decimal types can be decreased, logical fields can be changed to character, and integers can be changed to logical.

If you do not need the default conversion of data-types, enable the TGT_DATATYPE_OVERRIDE property. This property retains the value of a data-type as is and does not convert it to meet the target schema requirements. However, you must recheck the validations because they may fail.

Table and column names on the target database can be the same or different from those on the source database. If they are the same, the Pro2 Auto-map function is used. Otherwise, the tables are manually mapped, but, mapping must be done once and saved for use in subsequent implementations.

If you want to use literal table and column names for your target database (SQL only), then you can enable the TGT_USE_LITERAL_NAMES property. This property allows you to use the same naming convention as the source database for tables and columns without modifying them to meet the requirements of the target database. Another property GEN_SQL_DEC_FORMAT, is used to set the format for decimal fields in both schema and differential files. If the property is set to YES, the format is same as in the source database. If it is set to NO then the default format [decimal(17,2) null] is used. The default value for this property is NO.

Split replication threads

You can split each replication thread into multiple split threads regardless of whether it serves replication of a single table or group of tables. You can split a single thread into a maximum of ten split threads. When you split a thread into *n* number of parts, it provides *n*-1 split threads. For example, if you split the thread 5 (replBatch5.bat) into four parts, it provides three split threads 51, 52, and 53 along with the original thread 5. This means, the four parts are threads 5, 51, 52, and 53.

Pro2 determines the number of a split thread dynamically based on the RECID of the source record, and groups together all the split threads that belong to the same record or RECID. This allows you to run multiple processes without adding queue compression and prevents out-of-sequence processing of more changes to the same ROWID or RECID.

The following topics are covered in split replication threads:

- Generate new trigger code for splitting threads.
- Run split threads.
- Enable and disable queue compression.
- Generate trigger code in lower case.

Generate new trigger code for splitting threads

You can generate and install a new replication trigger code to implement thread splitting, or replace the old triggers for tables or threads that you want to split. It is recommended to replace all the existing trigger code with the newly generated trigger code using a split trigger template.

The following split trigger templates are available in the replication template folder `Pro2\bprepl\repl_tmpl`:

- **tplt_repltrig_split_without_compression.p**—Splits threads but does not apply queue compression.
- **tplt_repltrig_split_with_compression.p**—Splits threads and supports queue compression.

To generate new trigger code:

1. In the Pro2 web interface, navigate to the **Properties** tab.
2. Search for `DEL_TRIG_TEMPLATE` and click **Edit**.

   The **Edit Property** window appears.
3. Specify the Value as `tplt_repltrig_split_with_compression.p` or `tplt_repltrig_split_without_compression.p.` and click **Save**.
4. Search for `WRI_TRIG_TEMPLATE` and click **Edit**.

   The **Edit Property** window appears.
5. Specify the Value as `tplt_repltrig_split_with_compression.p` or `tplt_repltrig_split_without_compression.p.` and click **Save**.
6. In the Pro2 web interface, click the **Properties** tab and modify the property values of the templates `DEL_TRIG_TEMPLATE` and `WRI_TRIG_TEMPLATE` to point to one of the split trigger templates.
7. Click **Manage Replication** , and select the database instance.

   The **Edit Replication** window appears.
8. Click **Next**.
9. Navigate to the **Generate Code** tab, and select the **Replication Triggers** checkbox to regenerate the trigger code.

   Note: Check if the `jobrunner.bat` file is running. If not, navigate to `\bprepl\Scripts` from **Windows Explorer** , and double click the `jobrunner.bat` file. The **Proenv** window appears.

   ---
10. Check if new code is generated by navigating to the `\bprepl\repl_d` folder. The date and time of the files in that folder are recent if they were updated successfully.
11. Copy the `repl_d` and `repl_w` folders from `\bprepl`, and paste them in `\db\bprepl` folder.

Run the split replication threads function

To run the split replication threads function, consider the example of splitting the replication thread into four parts and perform the following steps:

Note: You can only run split threads after generating new trigger code.

1. Create three new `.bat` files:
   a. Create three copies of the `replBatch5.bat` file located in the `Pro2\bprepl\Scripts` folder and rename them as `replBatch51.bat`, `replBatch52.bat`, and `replBatch53.bat` respectively.
   b. Edit the `replBatch51.bat` file, locate and modify Thread=5 to Thread=51, and save your changes. Repeat this step for `replBatch52.bat` and `replBatch53.bat` files.
2. Create new thread positions:
   a. Go to the Pro2 web interface and click **Manage Replication**, and select the database instance.

      The Edit Replication window appears.
   b. Click the **Advanced Configuration** tab.
   c. For thread 51, type `51` in the field next to the **Create New Thread** button, and then click **Create New Thread**. Repeat this step for threads 52 and 53.
3. Create a new property:
   a. In the Pro2 interface, click the **Properties** tab, and then click **Add**.
   b. In the **Property Name** field, specify the name of the property as `SPLIT_THREAD5`.
   c. In the **Value** field, specify the number of threads as 4.

You can add up to *n*-1 value where *n* is the number of thread. The maximum value allowed per thread is 10. However, if you specify a value (characters or negative numbers) other than the numbers 1 to 10, then the trigger uses the thread number.

After generating the trigger code and creating new property for split threads, if you do not create the `.bat` files and do not schedule the task entries, then the data remains in the queue until the new thread jobs are started.

Progress recommends that you do not change the split number if there are existing records in the queue. Changing the split number might push the changes of the same record and row to a differently calculated queue.

Assign a table to the split thread

After the replication thread is split, you assign a table to the thread.

To assign a table to the thread:

1. In the Pro2 web interface, click **Manage Replication**, and select the database instance.

   The **Edit Replication** window appears.

2. Select **Advanced Configuration** and assign a table (for example, `Customer` table) to Thread 5 from the **Thread#** column.

3. Run a replication job to assign the replication queue to the split thread. For example, you could run the following query in the **Procedure Editor**:

```
DEFINE VARIABLE i AS INTEGER NO-UNDO
REPEAT i = 1 TO 1000:
CREATE customer
ASSIGN cust-num = NEXT-VALUE(next-cust-num)
       NAME = "test_" + STRING(i)
       country = "USA".
END.
```

4. Check that the threads have split in the Pro2 web interface by clicking the **Total Threads** watch-box. The threads are distributed among the split threads.

5. To execute the replication records, run the `replbatch5.bat, replbatch51.bat, replbatch52.bat` and `replbatch53.bat` files.

   After the replication is complete, the status of replication threads appear as running in the **Replication** section of the Pro2 web Dashboard.

Enable or disable queue compression for splitting threads

Note: This is an optional property and is specific to Pro2 WAN setup only.

You can enable or disable queue compression by means of the `APPSRV_QUEUE_COMPRESSION` property. You can add this property by manual intervention from the web UI.

1. Click **Properties** tab and select **Add**.
2. Enter the **Property Name** as `APPSRV_QUEUE_COMPRESSION`.
3. Set the **Value** to either **YES** or **NO**.
4. Click **Save**.

Generate trigger code in lower case

Note: This is an optional property.

You can generate the trigger names in lower case by setting the property `TRIGGER_FORCE_LOWERCASE` to `YES` in the Pro2 web interface.

1. From the Pro2 web interface, click **Properties** tab and select **Add**.
2. Enter the **Property Name** as `TRIGGER_FORCE_LOWERCASE`.
3. Set the **Value** to either **YES** or **NO**.
4. Click **Save**.

Configure Pro2 to target SQL database connectivity

To configure the OpenEdge DataServer for Microsoft SQL Server or Oracle, you must configure the Open Database Connectivity driver, create a schema holder, and create a target database.

Pro2 is designed to replicate data as one-way connection from the OpenEdge source database to the target database. To facilitate this one way connection, Pro2 assumes the target database is read-only. If you plan to make changes to your target database, be aware that these changes can be overwritten by Pro2 when a tables or row are updated one the source database.

Standard data-type conversions

Pro2 performs some data type conversions automatically to circumvent issues that would typically break other types of Open Database Connectivity (ODBC) data transfers from OpenEdge to a Microsoft SQL Server or Oracle database.

- **Date conversions**— Some versions of Microsoft SQL Server and Oracle databases cannot handle dates with a value before January 1, 1753. To work around this issue, Pro2 preserves the month and the date and sets the year to 1753.
- **Column width**—In OpenEdge databases, data is stored in variable length fields, regardless of the field width definition. However, in Microsoft SQL Server and Oracle databases, column width is fixed. This would cause the Microsoft SQL Server or Oracle database to have a larger data footprint than the source OpenEdge database. To work around this, Pro2 truncates data to the column width that is defined in the target schema.
- **Logical fields**—The OpenEdge database recognizes three possible values for logical fields: True (1), False (0), and Unknown (?). The value Unknown is not recognized in other databases. To work around this, if a logical value is True, then Pro2 sets the target field to True, otherwise it is set it to False.

Data types on the target side can be the same or different from the source database. For example, precision of decimal types can be decreased, logical fields can be changed to character, or integers can be changed to logical.

If you do not need the standard data-types conversions, you can enable the `TGT_DATATYPE_OVERRIDE` property. This property retains the value of a data-type and does not convert it to adhere to the target schema requirements. However, you must recheck the validations as they may fail.

Table and column names on the target database can be the same or different from those on the source database. If they are the same, the Pro2 auto-map function on the mapping page of the user interface can be used. Otherwise, the tables need to be manually mapped. However, table mapping is required only once and can be reused in subsequent implementations.

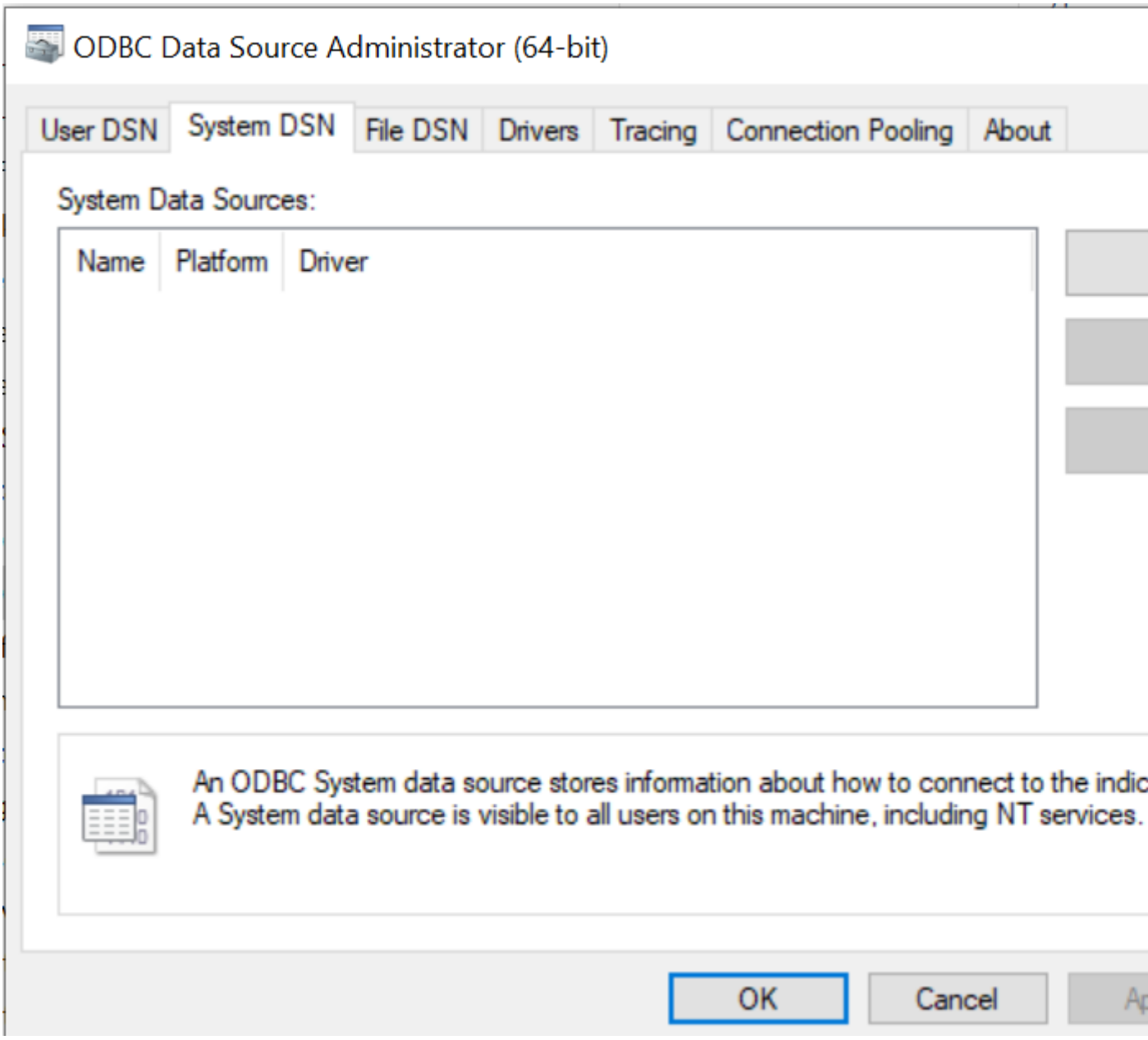
If you want to use literal table and column names for your target SQL database, enable the `TGT_USE_LITERAL_NAMES` property. This property allows you to use the same naming convention as the source database for tables and columns without modifying them to meet the requirements of the target database.

Configure the ODBC DataServer

The ODBC connectivity must be set up before you configure the OpenEdge DataServer. This allows communication between the OpenEdge database and another database.

To configure the ODBC DataServer:

1. Add a new system DSN using the **Windows Data Sources** tool by selecting the System DNS Tab in the **Data Sources (ODBC) Administrative Tools** section in the **Control Panel**.
2. Click **Add** to add a new system DSN.



3. For Microsoft SQL Server, choose the **SQL Server** driver. For Oracle, select the **Microsoft ODBC for Oracle** driver and click **Finish**.
4. Specify the name (for example *logical-db-name*ODBC) for your data source and the server where it is located, then click **Next**.
5. Provide your login information. If you are using Select SQL authentication, provide the login ID and password for the Microsoft SQL Server database, and click **Next**.
6. From the **Change the default database** to drop-down, select the appropriate database, and click **Next**.
7. Accept the defaults on the next dialog box, and click **Next**.
8. Click the **Test connection** button to confirm that you are connected.
9. Exit the **Data Source Administration** tool.

Create an OpenEdge schema holder

An empty OpenEdge database is created to act as the schema holder for the Microsoft SQL Server or Oracle database. The schema holder is the layer that allows ABL to interact with the Microsoft SQL Server or Oracle database as though it were a native OpenEdge database.

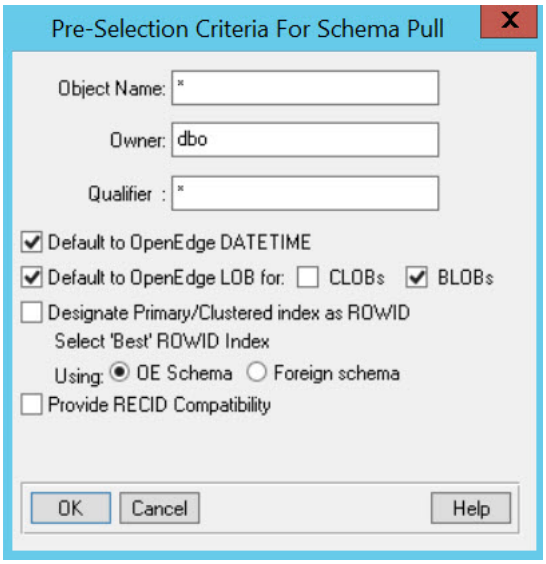
To create a schema holder:

1. Open a **Proenv** window, and type `prodb` *`database name`* `empty` to create an empty schema holder database.
2. Start a single user session by typing `prowin` *`database name`* `-1`.

   The **Procedure Editor** window appears.
3. From the **Procedure Editor**, select **Tools** > **Data Administration**.
4. Select **DataServer** > ***`database type`*** **Utilities** > **Create DataServer Schema** .

   The **Create/Modify Record for Schema** window appears.
5. Specify **Logical Database Name**.
6. If the source database is a SQL server, specify the **ODBC Data Source Name**. Click **Ok**.
7. Enter the **Login ID** and **Password** if required in the next dialog box and click **OK**.
8. Enter criteria for **Object Name**, **Object Owner** (typically, `dbo`), and **Qualifier**.
9. Select the **Default to OpenEdge DATETIME** option.
10. If you are using LOB fields in replication, ensure that the **Default to OpenEdge LOB** for **BLOBs** option is selected along with the CLOBs option. This forces the data server to automatically consider certain target side data types to be treated as LOBs.

    By selecting CLOB, `varchar(max)/nvarchar(max)` data types are automatically converted to CLOB. By selecting BLOB the data server is forced to consider all `varbinary(max)` as BLOB. On source fields that are character types, Pro2 may convert some types to `varchar(max)` when the source width is wider than the `MAX_CHAR_WIDTH` property setting. This conversion has the added benefit of avoiding row size limits for SQL Servers. However, `varchar(max)/nvarchar(max)` are not necessarily treated as CLOBS.



11. Click **OK** to finish.

Target Side Only Tables and Fields

The following tables are added to the target database for Pro2 use:

`Pro2SQL` – used by Pro2 to verify connection to target database.

`PRROWID` – Unique key that corresponds to the source Progress `ROWID`.

`Pro2created` – Date/timestamp of the record was first created.

`Pro2modified` – Date/timestamp of the record was last modified.

`Pro2SrcPDB` – Name of the source database.

`Progress_RECID` – Used by `Pro2ORA`. Required for `Pro2SQL` with source databases before OpenEdge release 10.2.

`Progress_RECID_Ident` – Required for `Pro2ORA` and `Pro2SQL` with source databases before OpenEdge release 10.2.

Indexes in the SQL Target

Pro2 generates a script to copy all source side indexes to the target database with the exception that any unique keys are translated as non-unique. The source side indexes are not used by Pro2 and are not required. Not creating or dropping unused or unnecessary indexes on the target side can optimize Pro2 performance as there will be no overhead for index maintenance.

One of the benefits of the `Pro2aql` table is the that it enables you to add indexes on the SQL target database. Most OpenEdge applications have an index structure for Online Transaction Processing (OLTP), however many reporting tools operate more efficiently with an index structure for Online Analytical Processing (OLAP). Adding indexes for OLAP to the SQL target database is useful and recommended, but there is one caveat. The only unique index allowed on the SQL target database is the index on the `PRROWID` field. If there are `UNIQUE` indexes other than the `PRROWID` index, then replication may fail due to a `UNIQUE` key constraint. There is little value of a `UNIQUE` index in an OLAP environment, so all additional indexes should not be `UNIQUE`. If unique index is required, it is strongly recommended that the `PRROWID` column be part of that index.

Create target Microsoft SQL Server or Oracle database

If you have not done so already, create a target database for you replicated data. The Target database can be either a Microsoft SQL server, Oracle, or OpenEdge database. Creating a target database will depend largely on they type of database you want to use and your development tooling. The following procedure lists the general steps to creating your target database.

1. In your database admin tool, like Microsoft SQL server Management Studio, create a new database and name it according to your needs.
2. Create a new database query, and load the schema file into the query.

For more information about creating a Microsoft SQL Server database, see Microsoft SQL documentation.

For more information about creating an Oracle database, see Oracle Database Documentation

You can now configure the replication process and proceed with the replication.

Configure Pro2 e-mail notifications

You can configure Pro2 to send e-mail notifications when a replication thread stalls or stops.

To do this, Pro2 compares the timestamp of the thread's most recent replication activity (a heartbeat) against a configurable time period, regardless of the HEARTBEAT_CHECK _MINUTES property. The default time period is 10 seconds. If the last recorded activity of the thread is more than 10 seconds from the time of comparison, then an e-mail notification is triggered.

Note: The heartbeat functionality (with e-mail alarm) is applicable for both replication and CDC administrator threads.

To set up Pro2 e-mail alarms:

1. Edit the predefs.i file located in the Pro2 root directory and define the following e-mail properties:
   - FROM-EMAIL-ADDR—The e-mail address of the sender of the e-mail alert
   - TO-EMAIL-ADDR—The e-mail address of the recipient of the e-mail alert
   - COMPANY-MAIL-SERVER—The SMTP mail server of the sender of the e-mail alert
2. On the Pro2 web interface **Properties** tab, designate a user name and password by using the **SMTP AUTH** field. The user name and password should be separated by a comma.
3. Create a new task in **Windows Task Scheduler** to periodically run a batch file called statusCheck.bat.

Remove the e-mail notification

StatusCheck sends only one e-mail notification per day by default.

To remove the restriction of one notification per day:

1. In the Pro2 web interface, go to the **Properties** tab.
2. Set the value of the ALERT_1_PER_DAY property to **NO**.

Set up third-party e-mail program

To use a third-party command-line e-mail program like Blat:

1. In the Pro2 web interface, go to the **Properties** tab.
2. Set the property USE_3RD_PARTY_EMAIL to **YES**.
3. Edit the mail.bat file located in bprepl/scripts, and modify the batch script to send the e-mail. For example, for Blat, use the following statement:

   ```
   blat -to %1 -subject %2 -body %3 -from %4 -server %5 -q
   ```

Note: The argument values (%1, %2) are variables that are populated by the smtpmail.p procedure, so you need to pass the appropriate variable to the arguments of your command-line e-mail program. For example, %1 denotes the e-mail recipient address, %2 denotes the subject of the e-mail, and so on. The mail.bat file contains remarks that explain the purpose of each the variables.

Use second-pass replication

Second pass replication can be used in two ways: to perform data warehouse auditing tasks or to replicate to a first pass and second pass target database.

Set up second-pass replication as audit

Second-pass replication creates a copy of the source table in the second-pass database and, inserts a record into this table each time there is a change in the source table (for `UPDATE`, `DELETE`, `INSERT`).

A new record is created in the target (or staging) table every time there is a change in the table, regardless of whether the change happens to the same field. The replicated table holds a new record for every change in the source table. A new property, `SECOND_PASS`, is added to the `predefs.i` file. This property supports second-pass replication and the default value of this parameter is set to `NO`. To start Second Pass Replication for a setup, this property should be set to `YES` for that specific setup.

To set up second-pass replication:

1. Create a copy of the Pro2 setup directory and rename it, appending it with `SecondPass`. For example, if your setup is in the folder `Pro2`, copy paste that folder and rename it to `Pro2_SecondPass`.

   Note: Copy all folders from the Pro2 setup directory to the SecondPass folder except for the `db` and `Pro2Web(PAS Instance)`folders.

2. Create a new folder named `db` in the newly created SecondPass folder. This `db` folder is created for the SecondPass target database.
3. Open a **Proenv** window and enter `cd C:\Pro2_SecondPass\db` to navigate to the `db` folder, for example, `cd C:\Pro2_SecondPass\db`
4. Create an empty target database from the **Proenv** window. Enter `prodb` *`target database name`* `empty`, for example, `prodb secondtgt empty`.
5. Start the target database from the **Proenv** window. Enter `proserve` *`target database name`* `-S` *`port number`* for example, `proserve secondtgt -S 6677`.
6. Load the schema generated from first pass into the target database. In the **Proenv** window enter `prowin` *`target database name`*, for example, `prowin secondtgt`.

   The **Procedure Editor** window appears.
7. Navigate to **Tools** > **Data Administration**.

   The **Data Administration** window appears.
8. Navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)**.

   The **Load Data Definitions** window appears.
9. Select the schema file as the input file, for example, `Sports-tgt.df`. Click **OK**.

   A success message appears after the load is complete.
10. Navigate to `C:\`*`Pro2 second pass folder`*`\bprepl\Scripts` folder, for example, `C:\Pro2_SecondPass\bprepl\Scripts`.
11. Right-click the **Pro2 - Editor** tool and select **Properties**.

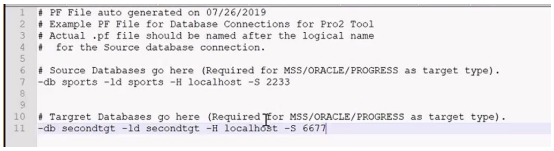
    The **Pro2 - Editor Properties** window appears.

12. On the **Shortcut** tab, edit the **Start in** field to reflect the *Pro2 second pass* folder, for example, C:\Pro2_SecondPass.

Note: You can use the same steps for the runbulkloader utility if the you need to do second pass bulk load.

---

13. Edit the Pro2_env.bat file to set the value of the Pro2 setup location to the C:\Pro2_SecondPass directory.
14. Open the replProc.pf file from C:/Pro2_SecondPass/bprepl/scripts folder and ensure that the rep and Pro2 database section to point to the first pass setup.
15. Save the contents of the replProc.pf file.
16. Edit your database's .pf file, and modify the target databases section to point to the database in the SecondPass directory. Save the contents of the .pf file. For example:



Note: If you use a Microsoft SQL Server or Oracle database, the name of the second pass schema holder is the same as the first pass and should reference a schema holder database that points to the new SecondPass target database.

---

17. From C:/Pro2_SecondPass, open the predefs.i file and change the value for SECOND_PASS to YES.
18. Set the first pass replication thread's disposition to mark as applied.

Note: This preserves the queue records so that the can be replicated on the second pass. As a consequence, you **must** run manual purges on the replqueue table.

---

19. From C:/Pro2_SecondPass/bprepl/repl_tmpl, rename the process library template name in second-pass setup directory.
    - For a WAN setup, rename the file tmpl_replproc_4audit.pto tmpl_as*TgtDb*.p.
    - For a LAN setup, rename the file tmpl_replproc_4SECOND_PASS.p to tmpl_replproc.p or current replication template.
20. Navigate to the Pro2 web interface and select **Properties**. Edit the WRI_TRIG_TEMPLATE to reflect tplt_repltrig-raw.p as **Value**. Click **Save**.
21. Navigate back to the **Properties** tab. Edit the DEL_TRIG_TEMPLATE to reflect tplt_repltrig-raw.p as the **Value**. Click **Save**.
22. From the \Pro2_secondpass\bprepl\scripts folder, double click the batchgen_GEN_PROCS.bat file to generate code.
23. Create and start the second pass replication threads from the c:\Pro2_SecondPass\bprepl\scripts folder.

Note: Alternatively, you can create second pass replication threads and control information by loading the `replbasev610-2ndPass.ini` file present in the Pro2 folder.

Set up second-pass replication non-audit

To setup second pass replication as non-audit, you follow similar steps as first pass replication. The second pass functions exactly the same as first pass in that when the records on the source database are modified, the changes are replicated to the first pass target, and then replicated to the second pass target. This method of replication always overwrites the existing record instead of creating new records.

To set up second pass replication in the non-audit configuration refer to and follow the same steps for second pass replication audit configuration up to step 19. Skip steps 19 to 22 and then resume at step 23.

Note: If you want to change a template for the second pass setup, you can change the property value from **Properties** tab in the user interface, and regenerate the processor libraries, triggers, or bulk copy in the second pass folder. To this, use the `batchgen_GEN_PROCS.bat` file in Scripts folder.

Additionally, you can reuse the same template from the first pass replication for the second pass as well for the replication write/delete trigger, and bulk copy/processor library templates. To bulk load the records to the second pass target, use the `C:\Pro2_secondpass\bprepl\scripts\RunBulkLoader` shortcut.

Bulk load with Pro2

Bulk loading is a process that can load large amounts of data into a target database in a relatively short period of time.

With bulk loading operations in Pro2, depending on the template used, data is replicated either one row at a time or multiple rows are loaded for a single transaction, resulting in a more efficient method to seed a database. Generally, bulk loading is done during the initial seeding of a database after Pro2 has been configured and after your target database is built.

The Bulk load utility can be run from the **Actions** tab on the Pro2 web interface. It can also be run from command line utility `bprepl\Scripts\runbulkloads.bat` by providing input parameter details. Both methods achieve the same result. The method that you choose depends on your preference and business needs.

Example of the command line syntax:

```
bprepl\RunBulkLoads.p
-param "DB=sports;TableInc=customer,
item;
TableExc=xxxxxxxxx;
RunFlag=A;
ForThread=0;
MaxRecordCount=0;
ResetBulkload=no;
ResetInProcess=no;
MProcThreads=9"
```

Note: The Bulk Load Utility always runs the standard bulk copy procedures in `bprepl\repl_mproc` folder in the Pro2 root directory.

Pro2 bulk load processor

The bulk load processor is a collection of programs that perform a bulk load for an entire table for an initial sync of the source table to the target table. A bulk load procedure executes a query such as `FOR EACH table-name: BUFFER-COPY table-name TO Target-Name`. This process can be run while replication is turned on. If deleted source database rows exist in the target, then they must be removed manually. To ensure that this does not happen, truncate the data from the target table before performing a bulk load.

Generate bulk load procedures

Before you generate bulk load procedures, ensure that you have configured your source and target databases, and mapped all tables that are to be loaded.

The bulk-load procedures can be generated from the **Manage Replication** window in the Pro2 web interface.

To generate bulk load procedures:

1. Select the **Manage Replication** tab, choose your database, and then click **Generate code**.
2. Select **Bulk-Copy Processor**. The bulk load procedures are generated in the directory specified by the `MASS_LOAD_DIRECTORY` property. The properties are set to `bprepl/repl_mproc` by default.

The template file specified by the `MASS_LOAD_TEMPLATE` property is used in the bulk load procedure generation. There are several templates for creating bulk load procedures. They are located in the `bprepl/repl_tmpl` directory and start with `tmpl_mreplproc`. Each of the templates have different functionality for different target types based on your needs. For example `tmpl_mreplproc_mssSendSQL.p` is for Microsoft SQL Server, while `tmpl_mreplproc_restart-auto-push.p` is used for any target type. If bulk load procedures that are generated using this template are stopped and then restarted, they continue from where they left off. In addition, if a source record is locked and cannot be copied, the procedures add the record to the `replqueue` to be written by the replication processor and the bulk load procedure moves on to the next record.

The `Bulk_Max_Cache` property tells the template how many processed row IDs to store in the cache. In general, if the loading fails or stops during the bulk load process and then restarted, the loading starts from the beginning. In such cases, the `Bulk_Max_Cache` property allows you to start the bulk load process from the last processed row IDs stored in the cache, instead of loading from the beginning. This property defaults to 25, and you can modify its value using the **Properties** tab in the Pro2 web interface.

For Oracle WAN implementations, ensure that the `APPSRV_MASS_LOAD_TEMPLATE` property uses the `tmpl_ASmproc_oracle.p` procedure, and the `APPSRV_FETCH_COUNT_MASS_BULK` property has the same value as the `Oracle_Bulk_Transaction_Count` property.

You can use the `INCLUDE_LOB` property to choose whether to include or exclude LOB data from replication and bulk loading. The property is set to `NO` by default, which excludes LOB data from replication and bulk loading. If the property is set to `YES`, then the LOB data is sent to the target database. This is applicable to both LAN and WAN environments.

The elapsed time information is provided in the log files generated by the bulk load procedures. You can find this information on each log entry per 100K rows and at the end of the loading process. Time elapsed is counted in seconds and milliseconds.

The bulk load procedures write to individual log files in the folder specified by the `MASS_LOAD_LOG_DIRECTORY` property (`bprepl\repl_mproclog` folder). The bulk load procedures can be initiated from the **Procedure Editor** or in a group by using the `mr source-database.p` procedure which runs each one in turn. Multiple bulk loads can also be launched at one time from the Bulk Load utility or the web user interface.

Table 2: Bulk load properties

Name	Default value	Description
APPSRV_MASS_LOAD_DIRECTORY	`bprepl/AppSrv/as_mproc`	This property designates the folder that Pro2 WAN implementations of Bulk Load procedures are generated.
APPSRV_MASS_LOAD_TEMPLATE	`tmpl_ASmproc.p`	Sets the name of the template procedure that is used during the bulk-load procedure for a WAN implementation. This property value can be set to: Default: `tmpl_ASmproc.p` Microsoft SQL Server: `tmpl_ASmproc_mssSendSQL.p` Oracle: `tmpl_ASmproc_oracle.p` Pro2 to Pro2 replication: `tmpl_ASmproc_pro2pro.p`
MASS_LOAD_TEMPLATE	`tmpl_mreplproc_restart-auto-push.p`	Sets the name of the template procedure that is used during the bulk-load generation.
MASS_LOAD_TEMPLATE_PRO2ORA	`tmpl_mreplproc_oracle.p`	This template is used in a LAN implementation for Oracle Target database. It uses the `send-sql` statement option to widen the Oracle transaction. The `send-sql` statement option does not support LOB fields.
MASS_LOAD_TEMPLATE_PRO2PRO	`tmpl_mreplproc_pro2pro.p`	This template is used in a LAN implementation with an OpenEdge Target database. It is faster than the default template, which includes the restart logic.
MASS_LOAD_TEMPLATE_PRO2SQL	`tmpl_mreplproc_mssSendSQL.p`	This template is used in LAN implementation for Microsoft SQL server. It uses the `send-sql` to perform faster but it does not support LOB fields.
MASS_LOAD_LOG_DIRECTORY	`bprepl/repl_mproclog`	Designates the Bulk Load log files storage folder location.
MASS_LOAD_DIRECTORY	`bprepl/repl_mproc`	This property designates the folder that Pro2 Bulk Load procedures are generated.

Run bulk load from the UI

After you have configured your source and target databases, mapped your tables, generated your bulk load procedure, and set your properties, you are ready to run a bulk load.

To run a bulk load:

1. Click the **Actions** tab and select **Bulk Load** > **Run Bulk Loads**.
2. Choose your **Source Database**.
3. (Optional) If you are rerunning bulk loads, reset your bulk load results.
4. Increase the **Run Flag** if more than one instance of a bulk load is run concurrently.

   Each instance must be given a separate flag (typically letters A through Z) to distinguish temporary files that are created and used by each instance. Each instance of the Bulk-Load Utility can have up to 5 bulk load threads. The number of bulk load utility instances and associated threads is limited by the Progress Pro2 license.

5. Set the **For mapped replication threads**

   Tables that are mapped to the replication thread entered are selected. Zero (0) specifies all threads.

6. Set the **Multi-thread bulk loads**

   The number bulk load processes or threads to run at one time. The default value is 9.

7. Indicate the **Max record count**. This property is used per table. Enter **0** to set a limitless maximum record count.
8. Set the **Re-load table if already bulk loaded once**

   Pro2 tracks the tables that have been bulk loaded and stores information in the `Repl_Control` tables. It uses this data to restart a load if it is stopped. It also skips loading a table that is already loaded. If you set this flag to `YES` it deletes any `COMPLETED` control records, so the table re-loads from the beginning.

9. Set the **Reset the "In Process"** if already bulk loaded once.

   Pro2 tracks the bulk load status and stores that information in `Repl_Control` tables. It uses this data to restart **In Process** tables if a load was stopped.

10. Set the **Table include list**

    A comma separated list of tables to include in the bulk load. Load patterns can be specified by using `*` as a wild card. For example: `ca*,da*,e*,z*`. This performs a bulk-copy of all tables starting with `ca`, `da`, `e`, and `z`.

11. Set the **Table exception list**

    A comma separated list of tables to exclude or skip. Load patterns can be specified using `*` as a wild card. For example, `ca*,da*,e*,z*`. This example performs a bulk-copy of all tables except those starting with `ca`, `da`, `e`, and `z`.

Run bulk loads from the command line

To run bulk loads from the `bprepl\Scripts` folder, run the `runblkld.bat` file from the `Scripts` folder and use the following syntax to run bulk loads:

Table 3: Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p %CODEDIR%\RunBulkLoads.p
 -param "ResetBulkload=no;ResetInProcess=no" >> %CODEDIR%\repl_mproclog\runbulkloads.log
```

Table 4: Parameters

Parameter	Description
`TableInc`	Comma-separated list of table names to be loaded. If not included on the parameter line, the program defaults to * (all tables).
`TableExc`	Comma-separated list of table names to be excluded from the load. If not included on the parameter line, the program defaults to xxxxxxx (no tables to be excluded).
`ResetBulkload`	Used to determine if any tables previously marked as complete should be reloaded. Valid entries are `YES` or `NO`.
`ResetInProcess`	Used to determine if the load should restart any tables with a status of `In process`. `YES` means restart from the beginning, and `NO` means to attempt to restart where the load previously stopped.
`DB`	This parameter should point to a mapped source database. It can be left blank, or you can remove it by using the `-params` line to load all mapped databases. Invalid entries cause the load to abort.
`RunFlag`	Determines which run flag to be used. This parameter defaults to `RunFlag=A`.
`ForThread`	If entered, the process loads only tables mapped to the `ForThread` value. This parameter defaults to `ForThread=0`. You can load tables for any thread number.
`MProcThreads`	This parameter is used to determine how many threads `runFlag` process will use for a load. The default for this parameter defaults `MProcThreads=9`. The maximum value is 9. Values greater than 9 will be reset to 9.
`MaxRecordCount`	This parameter is primarily used for testing. If a value is entered, the load for each table is stopped after the `MaxRecordCount` is reached. Default is `MaxRecordCount=0`.

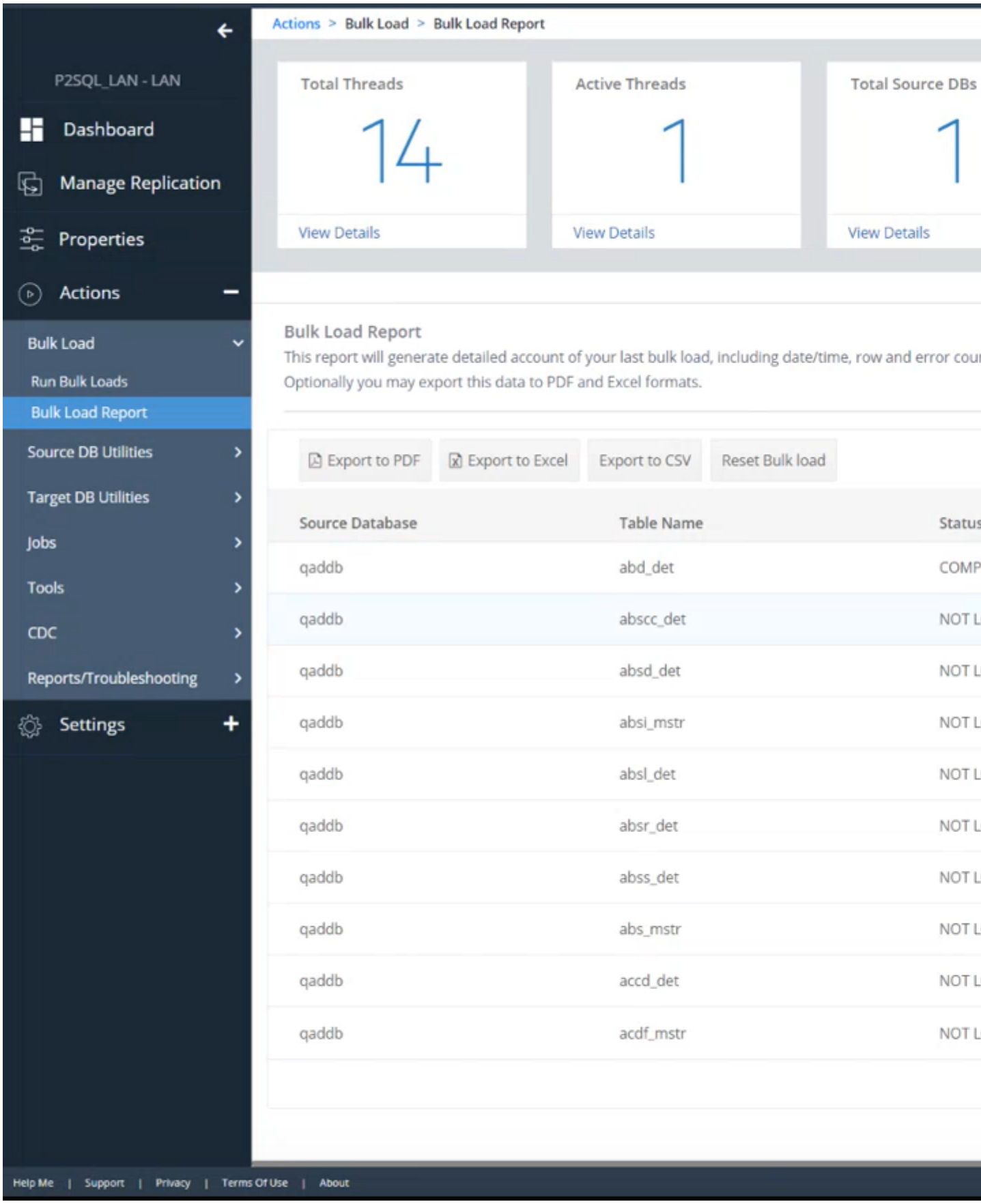
ASCII Bulk load and export

For WAN implementations, the ASCII bulk export/import is designed to sync replication tables from source to target. For large tables (typically over 50 million rows), the ASCII Bulk-export/load utility can be used to decrease the time to perform the bulk load. The process uses an ASCII dump export process on the source server. This process creates the dump and SQL load procedures, and then dumps the data to ASCII flat files. After this is finished, the dumped ASCII data files and the generated SQL load procedures are transferred to the target database server. The SQL load procedures are then run from the SQL query editor.

Bulk load reports

Generating a bulk load report can be useful in a variety of situations and can be essential for troubleshooting. A bulk load report can be generated in two ways: from the Pro2 user interface under **Action** > **Bulk Load** > **Bulk Load Report**, or by using the GET bulk load report API. For more information about how to use the API, see GET bulk load report API on page 100.

The bulk load report generates report of all mapped tables associated with your bulk load history. The bulk load report can be exported in PDF, Excel, and CSV formats. When the export job is complete, you can download the exported report in CSV format from the **Pending Jobs** watchbox.


Actions > Bulk Load > Bulk Load Report
P2SQL_LAN - LAN
Dashboard
Manage Replication
Properties
Actions
Bulk Load
Run Bulk Loads
Bulk Load Report
Source DB Utilities
Target DB Utilities
Jobs
Tools
CDC
Reports/Troubleshooting
Settings
Total Threads
14
View Details
Active Threads
1
View Details
Total Source DBs
1
View Details
Bulk Load Report
This report will generate detailed account of your last bulk load, including date/time, row and error cou
Optionally you may export this data to PDF and Excel formats.
Export to PDF
Export to Excel
Export to CSV
Reset Bulk load
Source Database
Table Name
Status
qaddb abd_det COMP
qaddb abscc_det NOT L
qaddb absd_det NOT L
qaddb absi_mstr NOT L
qaddb absl_det NOT L
qaddb absr_det NOT L
qaddb abss_det NOT L
qaddb abs_mstr NOT L
qaddb accd_det NOT L
qaddb acdf_mstr NOT L
Help Me | Support | Privacy | Terms Of Use | About

Dump and load with Pro2

Pro2 can help your OpenEdge implementation stay highly available by limiting down time during a dump and load procedure.

Index reorganization is the primary benefit of a dump and load. Usually, about 80 to 90 percent of the benefit of a dump and load can be achieved with index maintenance using the index rebuild and compact utilities. However, dumping and loading is still necessary to maintain performance standards. It can also be useful when you need to reorganize your database into different areas or extents due to changes in application requirements.

But back to Pro2. The message here is that downtime for dump and load processing can be limited to a few hours regardless of the database size. That is the entire point. It is not about the speed of the dump and load process as clearly a binary dump and load will be hundreds of times faster than a pro2 bulkload, but that is not the point. The point is this. If it takes 20 hours to perform a dump and load of a 500 GB database, using the fastest available method. If the business cannot tolerate a 20 hour outage, then you can use pro2 to replicate to another OpenEdge database. The bulkload may take a week, but that is not important. When you are ready to cut-over to the freshly bulkloaded database, the downtime will be a few hours only. The downtime is the same if the database is 10 GB, 100 GB, 1 TB, 10 TB. That is the value that Pro2 brings to the dump and load processing.

In general, there are two recommended dump and load methods for Pro2, the Data Dictionary dump and load tool and bulk loading.

For more information about the Data Dictionary dump and load tool, see Overview of dumping and loading a database.

For larger databases, use the Pro2 bulk load functionality, as it is the fastest method of migrating data from one database to another through Pro2. The bulk loader files are loaded sequentially with the indexes turned off, necessitating an index rebuild after the load completes.

For more information about migrating your database with Pro2, see Migrate your OpenEdge database with Pro2 on page 83

The point of using Pro2 to perform the dump and load process is to limit downtime for large databases in a high availability environment. Large databases will never use the dictionary dump and load. I am attaching a presentation I did a few years back on dump and load so you can see what performs best. This shows the dictionary dump and load being 8 times slower than the fastest method being binary dump and load.

Migrate your OpenEdge database with Pro2

You can use Pro2 to migrate your OpenEdge database with planned minimal downtime.

To stay competitive in the modern business world, your applications need to run as close to 24/7 as possible. In some cases, you can only have downtime limited to 4 hours. This can make upgrading or migrating your OpenEdge database difficult, even with proper planning. With Pro2 version 6.2, you can migrate your OpenEdge database with minimal downtime.

With proper planning, two short downtime events are needed to complete the migration process. The first event is used to setup the source database and to configure how changes to the database are recorded, whether they are trigger based or CDC.

The second downtime is used as the “cut over” event. The application continues to use the original database up to the point of the cut over.

A benefit of this method is if the project becomes nonviable for any reason, the original database is intact and functional. Downtime events, though small, can be planned to accommodate business specific priorities such as end of the month, end of quarter, etc.

To Migrate your OpenEdge database with Pro2, watch the following video, or use the procedure below.

The first step in the migration process is to prepare your source and target databases.

To prepare your source database for migration, you must embed Pro2 repl tables directly into your OpenEdge database.

1. Open a **Proenv** instance.
2. Navigate to the `Pro2` folder from the **Proenv** window. Type `cd` followed by the name of the folder where you installed Pro2, for example, `cd Pro2v61`.
3. Type `cd db` to navigate to the database folder of Pro2.

   Note: Check the directory of the `db` folder to view the database instances. Type `dir` after navigating to the database folder.

4. Copy the `repl.df` and `repl.st` files from the Pro2 installation folder.
5. Create an empty Repl database instance by typing `prodb repl empty`.
6. Load the required table definitions in the Repl database by typing `prowin repl -1`.

   The **Procedure Editor** window appears.
7. Navigate to **Tools** > **Data Administration**.
8. In the **Data Administration** window, navigate to **Admin** > **Load Data and Definitions** > **Data Definitions (.df file)…**.
9. In the **Load Data Definitions** window, load the `repl.df` file and click **OK**.

   Note: The `repl.df` file is located in *`Pro2 folder`*`/db`.

10. Start the source `repl` database instance by entering `proserve` *`database name`* `-S port number` in **Proenv**.

Now prepare you target database for migration:

1. Copy the `.st` and the `.df` files from your source database.
2. Modify the `.df` to be a type 2, target database.
3. Set up the new target database with a logical name so that replication program can identify it.

For the next procedure, go to the Pro2 UI and add a new replication:

1. From the Pro2 web interface, navigate to the **Manage Replication** tab.
2. Click **New**.

   The **Create Replication** window appears.
3. Set up global settings for the replication instance, for example, **Source DB Mode** is set to **Triggers**, and the **Source DB Connection** is set to **LAN**.
4. Click **Next**.

   The **Select Source** tab appears.
5. Enter **Source DB Name**
6. Enter **Host Name**
7. Enter the port number in the `Host Port/Service` field (for example, 2233).
8. (Optional) Enter your **User Name** and **Password**.

   Note: Enter your user name and password only if it is required by your source database.
9. Select **Test Connection**.

   If the test is successful, move on to set the target database. Review your source database details to ensure they are correct.
10. Click **Next**.

    The **Set Target** tab appears.

Add the target database details. The target database is where your replicated data flows to when a replication is initiated.

1. Set **OpenEdge** as the **Target Database Type**.
2. Enter the **Target Database Name** and **Logical Name**.
3. Set the **DB Host Name** IP address.
4. Enter the port number in the **DB Port/Service** field.
5. Next, **Generate the Target Schema** `.df` files.

   This `.df` files contain the changes needed to facilitate the database migration.
6. Copy and paste them on to your target database.

After copying the file into your target database, you can begin the target database configuration process.

1. Open the **Data Administrator** and use **Load Data and definitions** to load the target database `.df` file.
2. After the process is completed, verify that the Pro2 fields have populated.
3. Now load the inactive index `.df` file. This deactivates all of the running indexes.
4. When complete, verify that the indexes are inactive.

Return to Pro2 and finish the replication configuration process.

After replication is configured, go to the pro2 dashboard to start the bulk load process.

1. Go to **Actions**.
2. Click **Run bulk load**.
3. Select your OpenEdge source database.
4. Clear the bulk load results and click **Run**.

After the Bulk load process is completed, go to you target database and run the Procedure editor to confirm that the records were replicated.

It is essential that both bulk load and replication processes are running.

After enough time has passed and you are ready to complete the database migration, return to Pro2 and generate the sync-up `.df` file.

1. Go to **Actions**.
2. Select **Target Database Utilities**.
3. Click **Target Differential**.
4. Enter the **Source Database** name.
5. Select **OpenEdge Database Migration**, and click **Submit**.
6. After the job completes, copy and paste the sync `.df` file into your target database.
7. In the last step, you validate that there are no differences between your two databases by generating a CRC report.

You are now ready to start using your new database.

Check replication logs

The ReplLog Check feature allows you to scan all the log files for any errors and reports them through an email. To execute this feature, run the `RunReplLogChk.bat` file from the `\Pro2\bprepl\Scripts` folder.

If you are running this functionality for the first time, a `ErrorTrigger.lst` file is created (available in the `repl_log` folder). Add the error list that needs to be scanned/excluded in this file separated by commas. This feature scans all the log files, excluding the unwanted errors, and then send an error report in an email.

You can log the errors in the `ErrorTrigger.lst` file as shown below:

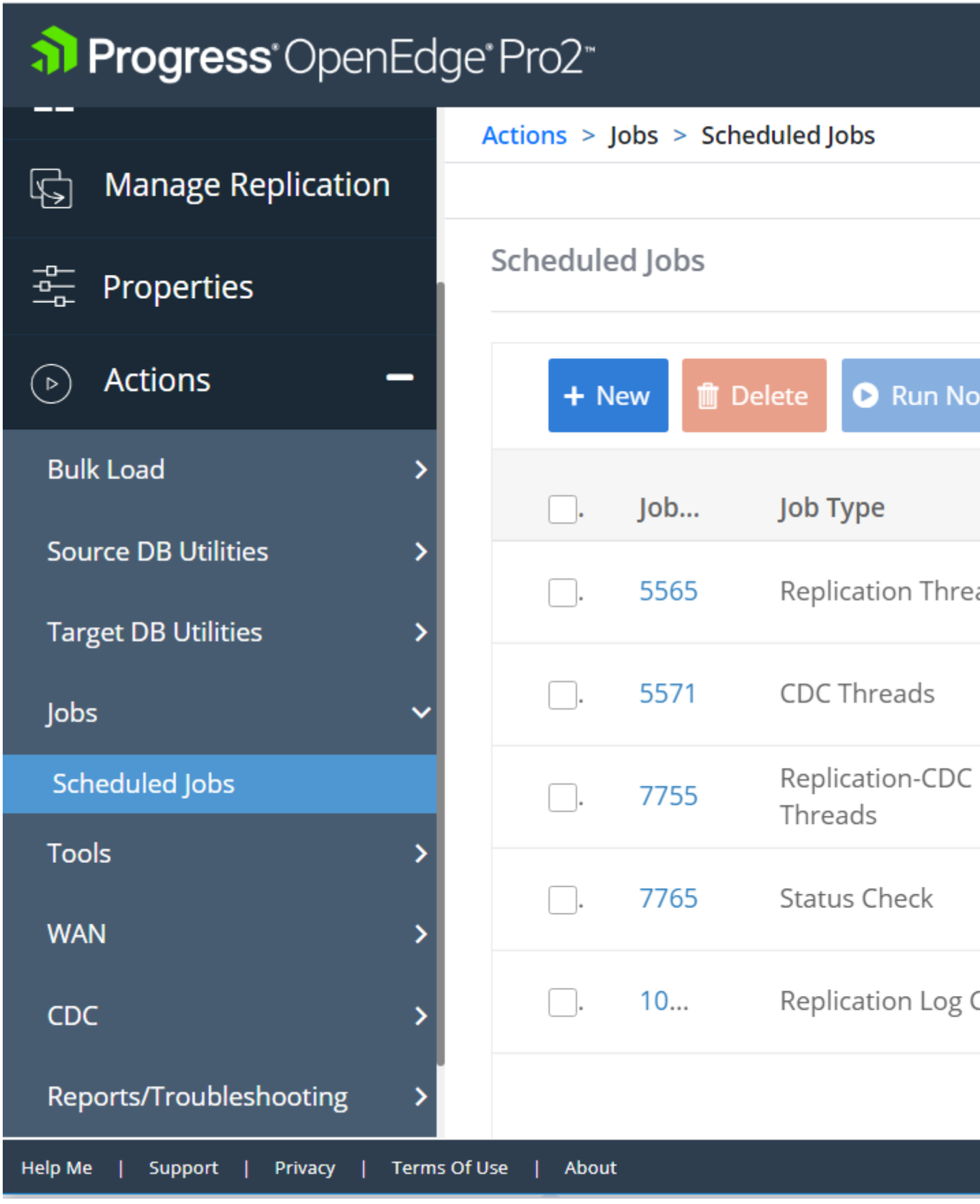
```
[ADMIN_LOG]
error,[SQL Server],
[ADMIN_LOG-EXCLUDES]
Warning
[REPL_LOG]
Error,[SQL Server],Maximum number of client connections,Application server connect
failure,Appserver did not connect,Transport resources unavailable,
[REPL_LOG_EXCLUDES]
Warning,Closing Replication Log,Re-Opened Replication Proc Log,Openining Replication
[APPSRV_LOG]
Error,[SQL Server],
[APPSRV_LOG_EXCLUDES]
Warning,Closing Replication Log
[BULK_LOAD_LOG]
Error,[SQL Server],
[BULK_LOAD_LOG_EXCLUDES]
Warning
```

Note: If the `ErrorTrigger.lst` file is not found in the expected directory, it is created in `repl_log` directory with an empty skeleton.

How to schedule jobs in Pro2

Pro2 is primarily used for replicating and synchronizing selected data and database objects from a source database to a target database. You can choose to manually perform your replications or schedule them to replicate automatically.

You can also schedule a variety of jobs by using the job scheduler. The job scheduler enables you to schedule recurring replications, CDC tasks, file purges, log checks, and more so that you no longer need to perform these tasks manually.


Progress OpenEdge Pro2
Manage Replication
Properties
Actions
Bulk Load
Source DB Utilities
Target DB Utilities
Jobs
Scheduled Jobs
Tools
WAN
CDC
Reports/Troubleshooting
Actions > Jobs > Scheduled Jobs
Scheduled Jobs
+ New
Delete
Job...
Job Type
5565
Replication Threa
5571
CDC Threads
7755
Replication-CDC Threads
7765
Status Check
10...
Help Me | Support | Privacy | Terms Of Use | About

To schedule a job:

1. From the **Actions** tab of the side menu, click **Jobs** > **Scheduled jobs** > **New**.

2. Under the **Type** drop-down list, select the one of the following job types:
   - Applied Queue Record Purge
   - CDC Purge
   - CDC Threads
   - Enterprise Push
   - File Purge
   - Replication Log Check
   - Replication Threads
   - Replication-CDC Threads
   - Status Check
3. Enter a **Description** for the job. A description helps you differentiate between similarly scheduled jobs.
4. If you are creating a new job, then ensure that the **Enable** check-box is selected. If you want to disable a scheduled job, but not remove a scheduled job, then you can do so by using this check-box.
5. In the **Setting** section, choose the job's start and end date as well as the start and end time.

   You can schedule a job to occur once, daily, weekly, or monthly depending upon your business needs.
6. Click **Save**.

Use CDC with Pro2

OpenEdge provides support for a feature called Change Data Capture (CDC). CDC is an OpenEdge RDBMS feature that identifies and captures data that changed in tables of a source database, as a result of `CREATE`, `UPDATE`, and `DELETE` (CUD) operations.

CDC is an industry term that describes the process of duplicating subsets of OLTP data in an external data source with a relatively up-to-date version of relational data. The OpenEdge implementation of CDC provides a flexible and scalable capture process to facilitate the data extraction, transformation, and eventually the loading of the data to an external data source.

Pro2 leverages two methods to implement CDC: CDC Thread and Replication-CDC Thread.

The first method, CDC Thread, is the standard implementation of this feature which replaces ABL triggers with CDC. The Pro2 standard CDC implementation supports a feature called `thread#0`. When this feature is used, the CDC Thread converts all `CDC_change_tracking` records to ReplQueue records, regardless of thread that the table is mapped to. This provides the convenience of having a single CDC Thread that converts all CDC records to replication queue records. The CDC replication process is faster than using ABL triggers and can be managed online.

The second method, Replication-CDC Thread, is a faster more efficient method of CDC replication. This method is achieved from bypassing the conversion of CDC records to replication queue records by writing changes directly from the source CDC database to the target database. This results in a replication performance speed increase and a decrease in lag.

Each type has advantages over the other. Replication-CDC Thread has the aforementioned speed, while CDC Thread can use compression and thread splitting. For example, CDC Thread can take multiple updates to a single record with the same `rowid` and compress it down into a single record to be processed. In contrast, Replication-CDC Thread processes a record for each transaction that occurs. The method that you choose depends on your business needs.

Pro2 use only Level 0 policies. Pro2 creates policies with the suffix `_pro2` so that it is easily recognized by the application, for example, *`tablename`*`_pro2`. If you create custom policies, ensure that `_pro2` is appended to each policy create for use with Pro2. Moreover, Pro2 does not replicate any changes that are generated from non-Pro2 policies. To determine whether a record is processed, Pro2 uses the `_user-misc` field in the `_change_tracking-table`. After the CDC record is processed, it either assigns the `_user-misc` value to `applied`, or it deletes the CDC record based on disposition value.

Limitations

To avoid read/write conflicts between replication thread types, there are restrictions around which threads can run simultaneously.

- A CDC Thread does not start if any of the same Replication-CDC thread(s) are running. For example, CDC Thread `thread#1` and Replication-CDC `thread#1` cannot run simultaneously. Likewise, Replication `thread#1` and Replication-CDC `thread#1` cannot run simultaneously.
- By design Replication-CDC cannot use split threads or queue compression.
- Second pass replication is not supported with Replication-CDC threads.

If you encounter an error derived from one of the above restrictions, you can delete or modify any threads from **Total Threads** watch-box on the Pro2 dashboard. In the **Total Threads** watch-box, select the problemed thread and click **Delete**.

Choose between standard CDC or Replication-CDC

Assume a scenario where a you are running standard CDC replication with 5 active threads. Your replication threads 1-4 are configured normally, thread 5 is a 3-way split thread, and 2nd pass replication is not in play. The thread number list would be 1,2,3,4,5,51 and 52. At this point, LAN or WAN is irrelevant to the scenario.

Split threads are not supported for threads of type Replication-CDC. This means the Replication-CDC thread type is not an option for replication of tables mapped to thread 5.

Threads 1-4 are potential candidates for flipping the thread type. For this example, we will pick thread 4 to convert to a Replication-CDC thread type.

On the web user interface, add a new thread 4 with type Replication-CDC. There is no need to modify the mapping, and it does not matter if single or multiple tables replicate under thread 4.

Ensure that CDC is not running for thread 4.

Note: If you are running one CDC thread to convert all `_cdc` rows, then you must rework your thread conversion strategy, and stop CDC conversions of thread 4.

Start a Replication-CDC thread only for thread #4.

If performance is improved over the standard CDC replication method, you should consider converting threads 1-3 to Replication-CDC.

If performance is not improved, then stop Replication-CDC Thread 4, and revert the CDC changes and restart standard replication thread #4.

As a safety precaution, set the disposition of the new thread to `mark as applied`. Doing so allows you to fallback if you the results of the scenario are not satisfactory.

Enable CDC on the source database

For a CDC-based database replication, you must configure the source database as CDC.

Note: The same versions of OpenEdge must be loaded on both the Pro2 machine and source database machine. For example, if the OpenEdge Database release is 12.0, then 12.0 must be loaded on the Pro2 machine.

If you are unsure if your database has already been enabled for CDC, you test it by opening the **Procedure Editor** > **Tools** > **Data Dictionary** > **Show Hidden Tables** > **_cdc-change-tracking**. If the CDC Tracking Table is there, then your database is enabled for CDC.

To enable CDC on the source database:

Note: This procedure can be run in Single User mode only.

---

1. Open the **Proenv** window.
2. Shut down your source database and PAS for OpenEdge instance.
3. Direct your **Proenv** session to your database location and enter the following command to add the CDC structure file to the source database:

   ```
   prostrct add database name db/cdcadd.st
   ```

   Note: The structure file will assign particular areas in the source database for the CDC function.

   ---

4. Enter the following command to enable CDC on source database:

   ```
   proutil database name -C enablecdc area ReplCDCArea indexarea ReplCDCarea_IDX
   ```

5. Restart the PAS for OpenEdge instance and the source database.
6. In the Pro2 web interface, select the source database from the **Manage Replication** tab.

   The **Edit Replication** window appears.

7. Change the **Source DB Mode** to CDC.

Note: After you have enabled CDC on the source database, configure the replication process for CDC based replication.

Add, remove, and customize CDC threads

You can perform CDC-based replication after your source database is CDC enabled. Pro2 treats the both replication threads and CDC threads the same, allowing you greater flexibility in your replication configurations. You can add, remove and customize threads as needed. Again, there are two types of CDC threads that you can select, Replication-CDC Threads and CDC Threads.

Perform the following steps to add replication threads:

1. On the Pro2 web user interface dashboard, select **Total Threads**.

   The **Total Threads** window appears.

2. Click **+Add**.

   The **Add New Thread** window appears.

3. Select either **Replication-CDC Threads** or **CDC Threads**.
4. Enter a **Thread ID**. If the thread type is CDC, select the database and other controls according to your needs..

   Note: It is best practice to set the Disposition to **Delete** as it improves performance.

5. Choose your database, if you have multiple source databases.
6. Click **Add**.
7. The CDC thread appears in the **Total Threads** list.

To delete threads:

1. On the Pro2 UI dashboard, select **Total Threads** watchbox.

   The **Total Threads** window appears.
2. Choose either **Replication-CDC Threads** or **CDC Threads**.
3. Click the check box beside the thread that you want to delete and select **Delete**.

   The **Delete Threads** window appears.

   Note: You can delete multiple threads by selecting multiple check boxes at the same time.

4. Click **OK**. The thread no longer appears on the **Total Threads** window.

To customize CDC threads:

1. On the Pro2 UI dashboard, select the **Total Threads** watch-box.

   The **Total Threads** window appears.
2. Choose either **Replication-CDC Threads** or **CDC Threads**.
3. Indicate the threads you want to update.
4. Customize the **Logging Level**, **Disposition**, and **Thread Control** as needed using the Actions drop down.

Map CDC

To configure CDC replication, you must map your source and target database after your CDC source database is enabled.

After you have enabled CDC on source database, a **CDC Mapping** tab appears in the **Manage Replication** window.

To enable CDC mapping:

1. From the **Manage Replication** window on the Pro2 web interface, select the **Source Database**.
2. Ensure that the **Source DB Mode** is **CDC**. Choose the **Test DB Connection** button. A message will indicate if CDC is enabled on the source database.
3. Click the **Mapping** option and ensure that all tables to be replicated have been mapped from source to target.
4. Select **CDC Mapping** .
5. Select the source tables to be mapped for CDC and click **Add**.
6. Select the **Policy Settings** for each of the **Source Tables** in the **Add / Activate Policy** column.

Note: Pro2 replication is currently designed to capture records the have changed. In CDC, this is known as a Policy. Activating a policy forces the OpenEdge database to immediately start tracking changes to the selected tables. Deactivating a policy forces the OpenEdge database to immediately stop tracking changes to the selected tables.

You can use the **Table List** functionality to specify specific tables to map by entering them as a comma separated list. This can be useful when you have many tables to sift through or a select few tables that you want to map. By entering in your list of tables, you can **Add**, **Activate**, **Inactivate** or **Delete** the CDC Policy.

Schedule CDC jobs

You can also schedule a variety of jobs by using the job scheduler. The job scheduler enables you to schedule recurring replications, CDC tasks, file purges, log checks, and more so that you no longer need to perform these tasks manually.

To schedule a job:

1. From the **Actions** tab of the side menu, click **Jobs** > **Scheduled jobs** > **New**.
2. Under the **Type** drop-down list, select the one of the following job types:
   - CDC Purge
   - CDC Threads
   - Replication-CDC Threads
3. Enter a **Description** for the job. A description helps you differentiate between similarly scheduled jobs.
4. If you are creating a new job, then ensure that the **Enable** check-box is selected. If you want to disable a scheduled job, but not remove a scheduled job, then you can do so by using this check-box.
5. In the **Setting** section, choose the job's start and end date as well as the start and end time.

   You can schedule a job to occur once, daily, weekly, or monthly depending upon your business needs.
6. Indicate the **Database** you want to run the thread against, as well as **Thread number**.
7. Click **Save**.

Split CDC threads

You can split each CDC thread into multiple threads regardless of whether the CDC thread serves the replication of a single table, or a group of tables. You can split a single thread into a maximum of 10 split threads. When you split a thread into *n* number of parts, it provides *n*-1 split threads. For example, if you split the thread 5 (`CDCBatch5.bat`) into four parts, it provides three split threads 51, 52, and 53 along with the original thread 5. This means, the four parts are threads 5, 51, 52, and 53.

To split CDC threads:

1. Create new `.bat` files. For example:
   a. Create three copies of the `replBatch5.bat` file located in the `Pro2\bprepl\Scripts` folder and rename them as `replBatch51.bat`, `replBatch52.bat`, and `replBatch53.bat` respectively.
   b. Edit the `replBatch51.bat` file, locate and modify Thread=5 to Thread=51, and save your changes. Repeat this step for `replBatch52.bat` and `replBatch53.bat` files.
2. Create new thread positions:
   a. Go to the Pro2 web interface and click the **Total Threads** watchbox.
   b. b. Click the **Add** button,
   c. In the **Thread ID** type 51, and fill out the remaining settings.
   d. Click **Add** to repeat this step for threads 52 and 53.
3. Create a new property:
   a. In the Pro2 interface, click the **Properties** tab, and then click **Add**.
   b. In the **Property Name** field, specify the name of the property as `CDCSPLIT_THREAD5`.
   c. In the **Value** field, specify the number of threads as `4`.

   Note: You can add up to *n*-1 value where *n* is the number of threads. The maximum value allowed per thread is 10. However, if you specify a value (characters or negative numbers) other than the numbers 1 to 10, the CDC uses only the actual thread number.

After generating the CDC code and creating new property for split threads, if you do not create the `.bat` files and do not schedule the task entries, then the data remains in the queue until the new replication thread is started.

Progress recommends that you do not change the split number if there are existing records in the queue. Changing the split number might push the changes of the same record and row to a differently calculated queue.

Enable or disable queue compression for CDC

Note: This is an optional property and is disabled by default. This property is not applicable for Replication-CDC Threads

You can enable or disable queue compression by means of the `CDC_QUEUE_COMPRESSION` property. You can edit this property from the web UI.

1. Click **Properties** tab and select the **CDC_QUEUE_COMPRESSION** property.
2. Click **Edit**.
3. Set the **Value** to either **YES** or **NO**.
4. Click **Save**.

Optimize CDC performance

You can optimize performance for both LAN and WAN configurations that use CDC by running CDC replication threads directly on the source database machine of your deployment.

There are two methods to optimize CDC replication. The method that you use depends on the configuration of your deployment.

For LAN configurations:

1. Go to `bprepl/scripts/` in the Pro2 root folder.
2. Locate the `Build_CDC_pkg.bat` file.
3. Run the `Build_CDC_pkg.bat` file directly on your source database machine.

For WAN configurations:

1. Go to `bprepl/scripts/` in the Pro2 root folder.
2. Locate the `CDCBatch.bat` file.
3. Update the `CDCBatch.bat` with your database name and thread number. For example, `database=`*`database name`*`, Thread=1`.
4. Run the `bprepl/scripts/CDCBatch.bat` directly on the on the source database machine of your WAN deployment.

For more information about CDC optimization, see *Docs/README_CDC_Build.txt* in the root folder of your Pro2 installation.

Manage Pro2 Properties

You can manage Pro2 to properties by loading a properties file, also known as configuration file, into the Pro2 interface. After the properties are loaded, you can Add, Edit or Delete them using the interface.

All of the preset properties necessary for Pro2 functionality are stored in the `Repl_Properties` table of the Pro2 database. In the `Repl_Properties` table, you can configure settings for various properties, including the log file location, logical delete table parameters, procedure template specifications, and more.

Pro2 Configuration settings can also be saved to, and loaded, from a text file, and added to Pro2 by using the **New** button in the **Manage Replication** window on the Pro2 web interface.

To load and manage Pro2 properties:

1. On the Pro2 dashboard menu, click **Actions** > **Tools** > **Load Configuration**.
2. Click on select files button and choose the `replbasev610.ini` file for the latest configuration of v6.2 under Pro2 root folder. If you want to use Replication-CDC or second pass replication then, also select one of the following to load in addition to the `replbasev610.ini` file.

File name	Description
replbasev610-2ndPass.ini	`.ini` file for second pass replication.
replbasev610-Replication-CDC.ini	`.ini` for CDC based replication with the Replication-CDC thread type.

Note: You can only select one or the other. Due to the fundamental differences in indexing, second pass and Replication-CDC are not compatible with each other.

---

3. After choosing and loading your files, click the **Submit** button.

Note: After the configuration file is loaded, additional configuration repositories are created under multiple Pro2 database tables:

- Repl_Properties- Properties information
- Repl_Control- Control record information
- ThreadControl- Thread level information
- AlertType- Alert related information
- JobTemplate- Job template related information (used for Job scheduler)
- ReplP_wbdef- watch boxes related information.

If the **Overwrite Pro2 Properties** check box is selected, then all previous properties are overwritten based on property name.

To add properties from the web interface:

1. On the Pro2 dashboard menu, click **Properties** > **+Add**.
2. Enter the new property **Name** and **Value**.
3. Click **Save** to save and add the new property.

To edit a property that is already loaded to the interface:

1. On the Pro2 dashboard menu, click **Properties**.
2. Select the property that you want to edit and click **Edit**.
3. Enter the updated property **Name** and **Value**.
4. Click **Save** to save and update the property.

To delete a property using the web interface:

1. On the Pro2 dashboard menu, click **Properties**.
2. Select the property that you want to delete and click **Delete**.

To download a copy of the properties file from the Pro2 web interface:

1. After you have modified or added properties, the changes are enacted by the job runner. You can view the status of the change in the **Pending Jobs** watch box.
2. Click the **Pending Jobs** watch box.
3. Select the properties job that correlates to your recent property changes and click on the download icon. The `properties.ini` is downloaded.

GET bulk load report API

Call `GET bulkloadrpt` with your Pro2 log-in credentials to get a JSON report for a bulk load process that you specify as query parameters in the API call.

About this API

The GET bulk load report API is a REST API that, when called, queries a Pro2 database for a data report of a bulk load job. You specify the database host name and port in the URL of the call. To ensure security, you include the log-in credentials for your account in the query. You can use this API to check the progress of a bulk load process when initially seeding a database.

Request

Direct the API call to the URL:

```
http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/getbulkloadrpt&pVars=
```

Request format example as a curl command:

```
curl
-X GET
-u username:password
-v http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/getbulkloadrpt&pVars=
```

Response

The response returns a data-set report of the bulk load job in JSON format.

The following is an example response:

```
{
  "ProDataSet": {
    "ttRpt": [
      {
        "ttDB": "sports",
        "ttFileName": "Benefits",
        "ttTbl": "Benefits",
        "ttStatus": "COMPLETE",
        "ttDateTime": "01/28/2020-01:23:00",
        "ttRowsLoaded": 21,
        "ttRowsLocked": 0
      },
      {
        "ttDB": "sports",
        "ttFileName": "BillTo",
        "ttTbl": "BillTo",
        "ttStatus": "COMPLETE",
        "ttDateTime": "01/28/2020-01:23:01",
        "ttRowsLoaded": 2,
        "ttRowsLocked": 0
      },
      {
        "ttDB": "sports",
        "ttFileName": "Bin",
        "ttTbl": "Bin",
        "ttStatus": "COMPLETE",
        "ttDateTime": "01/28/2020-01:23:04",
        "ttRowsLoaded": 770,
        "ttRowsLocked": 0
      },
      {
        "ttDB": "sports",
        "ttFileName": "Customer",
        "ttTbl": "Customer",
        "ttStatus": "COMPLETE",
        "ttDateTime": "01/28/2020-01:23:04",
        "ttRowsLoaded": 1118,
        "ttRowsLocked": 0
      },
      {
        "ttDB": "sports",
        "ttFileName": "Department",
        "ttTbl": "Department",
        "ttStatus": "COMPLETE",
        "ttDateTime": "01/28/2020-01:23:03",
        "ttRowsLoaded": 7,
        "ttRowsLocked": 0
      },
      {
        "ttDB": "sports",
        "ttFileName": "Employee",
        "ttTbl": "Employee",
        "ttStatus": "COMPLETE",
        "ttDateTime": "01/28/2020-01:23:04",
        "ttRowsLoaded": 55,
        "ttRowsLocked": 0
      },
      {
        "ttDB": "sports",
        "ttFileName": "Family",
        "ttTbl": "Family",
        "ttStatus": "COMPLETE",
        "ttDateTime": "01/28/2020-01:23:04",
        "ttRowsLoaded": 72,
        "ttRowsLocked": 0
      },
      {
        "ttDB": "sports",
        "ttFileName": "Feedback",
```

```
    "ttTbl": "Feedback",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:05",
    "ttRowsLoaded": 8,
    "ttRowsLocked": 0
  },
  {
    "ttDB": "sports",
    "ttFileName": "InventoryTrans",
    "ttTbl": "InventoryTrans",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:06",
    "ttRowsLoaded": 75,
    "ttRowsLocked": 0
  },
  {
    "ttDB": "sports",
    "ttFileName": "Invoice",
    "ttTbl": "Invoice",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:06",
    "ttRowsLoaded": 147,
    "ttRowsLocked": 0
  },
  {
    "ttDB": "sports",
    "ttFileName": "Item",
    "ttTbl": "Item",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:07",
    "ttRowsLoaded": 55,
    "ttRowsLocked": 0
  },
  {
    "ttDB": "sports",
    "ttFileName": "LocalDefault",
    "ttTbl": "LocalDefault",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:08",
    "ttRowsLoaded": 10,
    "ttRowsLocked": 0
  },
  {
    "ttDB": "sports",
    "ttFileName": "Order",
    "ttTbl": "Order",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:14",
    "ttRowsLoaded": 3953,
    "ttRowsLocked": 0
  },
  {
    "ttDB": "sports",
    "ttFileName": "OrderLine",
    "ttTbl": "OrderLine",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:26",
    "ttRowsLoaded": 13970,
    "ttRowsLocked": 0
  },
  {
    "ttDB": "sports",
    "ttFileName": "POLine",
    "ttTbl": "POLine",
    "ttStatus": "COMPLETE",
    "ttDateTime": "01/28/2020-01:23:18",
    "ttRowsLoaded": 5337,
    "ttRowsLocked": 0
  },
```

```
{
  "ttDB": "sports",
  "ttFileName": "PurchaseOrder",
  "ttTbl": "PurchaseOrder",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:13",
  "ttRowsLoaded": 2129,
  "ttRowsLocked": 0
},
{
  "ttDB": "sports",
  "ttFileName": "RefCall",
  "ttTbl": "RefCall",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:11",
  "ttRowsLoaded": 13,
  "ttRowsLocked": 0
},
{
  "ttDB": "sports",
  "ttFileName": "Salesrep",
  "ttTbl": "Salesrep",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:12",
  "ttRowsLoaded": 9,
  "ttRowsLocked": 0
},
{
  "ttDB": "sports",
  "ttFileName": "ShipTo",
  "ttTbl": "ShipTo",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:12",
  "ttRowsLoaded": 3,
  "ttRowsLocked": 0
},
{
  "ttDB": "sports",
  "ttFileName": "State",
  "ttTbl": "State",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:13",
  "ttRowsLoaded": 51,
  "ttRowsLocked": 0
},
{
  "ttDB": "sports",
  "ttFileName": "Supplier",
  "ttTbl": "Supplier",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:14",
  "ttRowsLoaded": 10,
  "ttRowsLocked": 0
},
{
  "ttDB": "sports",
  "ttFileName": "SupplierItemXref",
  "ttTbl": "SupplierItemXref",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:14",
  "ttRowsLoaded": 56,
  "ttRowsLocked": 0
},
{
  "ttDB": "sports",
  "ttFileName": "TimeSheet",
  "ttTbl": "TimeSheet",
  "ttStatus": "COMPLETE",
  "ttDateTime": "01/28/2020-01:23:15",
```

```
          "ttRowsLoaded": 25,
          "ttRowsLocked": 0
        },
        {
          "ttDB": "sports",
          "ttFileName": "Vacation",
          "ttTbl": "Vacation",
          "ttStatus": "COMPLETE",
          "ttDateTime": "01/28/2020-01:23:16",
          "ttRowsLoaded": 12,
          "ttRowsLocked": 0
        },
        {
          "ttDB": "sports",
          "ttFileName": "Warehouse",
          "ttTbl": "Warehouse",
          "ttStatus": "COMPLETE",
          "ttDateTime": "01/28/2020-01:23:16",
          "ttRowsLoaded": 14,
          "ttRowsLocked": 0
        }
      ]
    }
}
```


GET replication queue records API

Call `GET threaddata` with your Pro2 log-in credentials to get a JSON report for replication queue records associated with specific threads.

About this API

The GET replication queue records API is a REST API that, when called, queries a Pro2 replication database for a list of replication queue records that are associated with specific threads in a replication database. You specify the database host name and port in the URL of the call. To ensure security, you include the log-in credentials for your account in the query. You can use this API to get a detailed report of the threads in a replication database, and up to 10 records associated with those threads.

Request

Direct the API call to the URL:

```
http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/threaddata
```

Request format example as a curl command:

```
curl
-X GET
-u username:password
-v http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/threaddata
```

Response

The response returns a data-set report of the replication queue records in JSON format under the `ttReplQueue` array.

The following is an example response:


```
{
  "ProDataSet": {
    "THREADCTL": [
      {
        "ThreadID": 0,
        "Updtime": "2020-04-02T03:33:30.444",
        "ThreadStatus": "STOPPED",
        "LogLevel": "Moderate",
        "Disposition": "Delete Record",
        "RepControl": "Enabled",
        "ThreadType": "CDC",
        "ControlType": "",
        "isAdd": false,
        "isRunning": false,
        "ttReplQueue": [
          {
            "Sequence": 1,
            "EventType": "C",
            "SrcDB": "sports",
            "SrcTable": "Customer",
            "SrcRecord": "0x0000000000000a10",
            "EventDate": "2020-04-02",
            "EventTime": "03:33:47",
            "SrcTransID": 0,
            "Username": "Administrator",
            "Applied": false,
            "ApplDate": null,
            "ApplTime": "",
            "Audited": false,
            "AudDate": null,
            "AudTime": "",
            "UserCust": "Administrator",
            "RawData": "",
            "QThread": 0,
            "THREADTYPE": "CDC"
          },
          {
            "Sequence": 2,
            "EventType": "W",
            "SrcDB": "sports",
            "SrcTable": "Customer",
            "SrcRecord": "0x0000000000000a10",
            "EventDate": "2020-04-02",
            "EventTime": "03:33:47",
            "SrcTransID": 0,
            "Username": "Administrator",
            "Applied": false,
            "ApplDate": null,
            "ApplTime": "",
            "Audited": false,
            "AudDate": null,
            "AudTime": "",
            "UserCust": "Administrator",
            "RawData": "",
            "QThread": 0,
            "THREADTYPE": "CDC"
          },
          {
            "Sequence": 3,
            "EventType": "C",
            "SrcDB": "sports",
            "SrcTable": "Customer",
            "SrcRecord": "0x0000000000000a11",
            "EventDate": "2020-04-02",
            "EventTime": "03:33:47",
            "SrcTransID": 0,
            "Username": "Administrator",
            "Applied": false,
```

```
            "ApplDate": null,
            "ApplTime": "",
            "Audited": false,
            "AudDate": null,
            "AudTime": "",
            "UserCust": "Administrator",
            "RawData": "",
            "QThread": 0,
            "THREADTYPE": "CDC"
          }
      }
    ]
  }
}
```


GET thread data API

Call `GET threaddata` with your Pro2 log-in credentials to get a JSON report for all data on specific threads.

About this API

The GET thread data API is a REST API that, when called, queries a Pro2 replication database for all data for specific threads in a replication database. You specify the database host name and port in the URL of the call. To ensure security, you include the log-in credentials for your account in the query. The request returns a variety of data, such as thread status, logging level, disposition, and more.

Request

Direct the API call to the URL:

```
http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/threaddata
```

Request format example as a curl command:

```
curl
-X GET
-u username:password
-v http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/threaddata
```

Response

The response returns a data-set report of all thread data in JSON format.

The following is an example response:


```
{
  "ProDataSet": {
    "THREADCTL": [
      {
        "ThreadID": 0,
        "Updtime": "2020-01-22T02:36:22.322",
        "ThreadStatus": "STOPPED",
        "LogLevel": "Moderate",
        "Disposition": "Delete Record",
        "RepControl": "Enabled",
        "ThreadType": "CDC",
        "ControlType": "",
        "isAdd": false,
        "isRunning": false
      },
      {
        "ThreadID": 0,
        "Updtime": "2020-01-29T05:02:54.958",
        "ThreadStatus": "STOPPED",
        "LogLevel": "Moderate",
        "Disposition": "Mark as Applied",
        "RepControl": "Enabled",
        "ThreadType": "JOB",
        "ControlType": "",
        "isAdd": false,
        "isRunning": false
      },
      {
        "ThreadID": 1,
        "Updtime": "2020-02-07T04:20:04.654",
        "ThreadStatus": "STOPPED",
        "LogLevel": "Minimum",
        "Disposition": "Mark as Applied",
        "RepControl": "Enabled",
        "ThreadType": "REPLICATION",
        "ControlType": "",
        "isAdd": false,
        "isRunning": false
      },
      {
        "ThreadID": 1,
        "Updtime": "2020-03-20T07:17:21.075",
        "ThreadStatus": "STOPPED",
        "LogLevel": "Moderate",
        "Disposition": "Delete Record",
        "RepControl": "Enabled",
        "ThreadType": "REPLICATION-AUDIT",
        "ControlType": "",
        "isAdd": false,
        "isRunning": false
      }
    ]
  }
}
```

GET Pro2 connection information API

Call `GET getconnection` with your Pro2 log-in credentials to get a JSON report for the type of connection that a Pro2 instance is using.

About this API

The GET connection information API is a REST API that, when called, queries a Pro2 instance the connection type that the instance is using. You specify the instance host name and port in the URL of the call. To ensure security, you include the log-in credentials for your account in the query. The request returns the connection type for the instance. The connection type can be LAN or WAN.

Request

Direct the API call to the URL:

```
http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/getconnection
```

Request format example as a curl command:

```
curl
-X GET
-u username:password
-v http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/getconnection
```

Response

The response returns a data-set report of connection information in JSON format.

The following is an example response:

```
{
  "ProDataSet": {
    "ack": [
      {
        "ConnType": "LAN",
        "AppSrvName": "",
        "AppSrvConnString": ""
      }
    ]
  }
}
```

GET version information API

Call `GET getaboutdata` with your Pro2 log-in credentials to get a JSON report of the version of a Pro2 instance.

About this API

The GET version information API is a REST API that, when called, queries a Pro2 instance for the version release that it is using. You specify the instance host name and port in the URL of the call. To ensure security, you include the log-in credentials for your account in the query. The request returns version information, for example, OpenEdge Pro2 Version 6.1.

Request

Direct the API call to the URL:

```
http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/getaboutdata
```

Request format example as a curl command:

```
curl
-X GET
-u username:password
-v http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/getaboutdata
```

Response

The response returns a data-set report of version information in JSON format.

The following is an example response:

```
{
  "ProDataSet": {
    "ttAbout": [
      {
        "ttTitle": "OpenEdge PRO2 Version 6.1",
        "ttDate": "Build: Tue, 11 Feb 2020 05:13:30",
        "ttVersion": "Version: 6.1"
      }
    ]
  }
}
```

GET instance information API

Call `GET repldbxref` with your Pro2 log-in credentials to get a JSON report of triggers, CDC information, and source and target database information for a Pro2 database.

About this API

The GET thread data API is a REST API that, when called, queries a Pro2 instance for a all data on that instance and that instances database. You specify the instance host name and port in the URL of the call. To ensure security, you include the log-in credentials for your account in the query. The request returns a variety of data, such as trigger data, CDC information, source and target database information, and more.

Request

Direct the API call to the URL:

```
http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/repldbxref&pVars=
```

Request format example as a curl command:

```
curl
-X GET
-u username:password
-v http://host_name:port/pro2/rest/Pro2/DPR?dMode=System/repldbxref&pVars=
```

Response

The response returns a data-set report of all thread data in JSON format.

The following is an example response:


```
{
  "ProDataSet": {
    "ttReplDBXref": [
      {
        "SrcDB": "sports",
        "SchHldr": "",
        "SchImg": "",
        "TgtType": "OpenEdge",
        "TgtConnName": "",
        "TgtPhysName": "target",
        "GenQRec": true,
        "ProcQRec": true,
        "SrcPhysName": "",
        "SchPhysName": "",
        "SrcConnectType": "LAN",
        "SrcDBMode": "CDC",
        "SrcHostName": "localhost",
        "SrcHostPort": "1122",
        "SrcUserName": "",
        "SrcUserPWD": "",
        "SrcLastJobID": 0,
        "AppsrvName": "",
        "TgtUserName": "",
        "AppsrvConnectStr": "",
        "TgtUserPWD": "",
        "TgtDBHost": "localhost",
        "TgtDBPort": "2233",
        "TgtLastJobID": 3562,
        "SrcDBPath": "",
        "TgtDBPath": "",
        "SchDBPath": ""
      }
    ]
  }
}
```

III

Additional information

For details, see the following topics:

- Pro2 FAQs
- Command-line references
- Processor procedure generator
- Pro2 repl databases and table schemas
- Pro2 directory hierarchy
- Pro2 program files
- Replication procedure library
- Replication processor
- Plan for disaster recovery with Pro2

Pro2 FAQs

When do I need to regenerate code?

You need to regenerate the replication code whenever there are mapping changes. However, triggers and bulk load processor code need to be generated only when a new table is added to replication. Replication processor library code needs to be regenerated whenever a table or field is added or removed.

When do I need to update table or field mapping?

The mapping of tables and fields does not update automatically when schema changes are made to either the source or target database. Once the changes are made to the target SQL schema and after the schema holder is updated, you need to update the mapping to map new tables/fields and to delete mapping of removed tables/fields.

When do I need to update the schema holder?

Anytime a change is made to the target SQL database schema the schema holder database needs to be rebuilt or updated.

How do I contact Pro2 Technical Support?

To report a new issue or update an existing issue, please login to the Progress SupportLink application at http://progresslink.progress.com/supportlink.

If you do not have a SupportLink login, please register at http://progresslink.progress.com/, or if you need immediate assistance please call us directly.

See the support contact page at http://web.progress.com/en/support/contact-support.html

A SupportLink login provides you with the ability to receive exclusive access to the SupportLink Web portal—a single location to access the latest support information, search our knowledge base and manage your support issues 24x7. SupportLink includes:

- Automated knowledge base searches during support case submission to find potential solutions to your issue.
- The ability to set case severity level and provide additional details on your business impact to help us quickly resolve your issue.
- The ability to define and store multiple personalized environments to associate with your support case at the click of a button.
- The ability to manage all your support cases by easily opening, closing and escalating issues.

Command-line references

Pro2 supports command-line operations for some of the web interface tool operations. This is an alternative to performing the tasks using the command prompt instead of the web interface. This topic explains the different modes and their respective syntax to execute the administrative tool operations. These operations can be executed using the `BatchGen.bat` file located in the `Scripts` folder of your Pro2 installation.

`LOAD_CONFIG`: Loads the configuration file from the location given using the `PATH` variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\replbasev4.ini;DB=All;Mode=LOAD_CONFIG " >> %CODEDIR%\repl_log\batch.log
```

`SAVE_CONFIG`: Saves the current configuration to the path given in the `PATH` variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= SAVE_CONFIG" >> %CODEDIR%\repl_log\batch.log
```

AUTO_DBXREF: Adds the database map record to the Database Mapping table.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"Mode=AUTO_DBXREF;DB=sports,Options=MSS >> %CODEDIR%\repl_log\batch.log
```

GEN_TGT: Adds the Microsoft SQL Server database map record to the Database Mapping table.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_TGT Options=NO|YES|NO " >> %CODEDIR%\repl_log\batch.log
```

AUTO_MAP: Maps the tables that were automatically mapped, listed in the Mapping table.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= AUTO_MAPP " >> %CODEDIR%\repl_log\batch.log
```

GEN_PROCS: Generates the Process libraries, Replication triggers, Bulk-copy processors to the default location specified in properties file.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_PROCS " >> %CODEDIR%\repl_log\batch.log
```

GEN_TGT_DIFF: Generates the target differential files and saves them to the location given in PATH variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_TGT_DIFF;Options=NO|YES|NO " >>
%CODEDIR%\repl_log\batch.log
```

TGT_DELTA_DF: Generates the trigger Delta DF file to the location given in the PATH variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= TGT_DELTA_DF " >> %CODEDIR%\repl_log\batch.log
```

GEN_DELTA_DF: Generates a Delta DF file on the WAN side and saves the files to the location given in the PATH variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_DELTA_DF " >> %CODEDIR%\repl_log\batch.log
```

CHECK_GEN_Q REC: Enables the functionality of generating replication queue records for single and multiple tables (which are separated by comma (,)).

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode=CHECK_GEN_QREC;TABLE=Customer" >> %CODEDIR%\repl_log\batch.log
```

Parameters

TABLE

Name of the mapped table. You can specify a single table name, multiple table names or ALL.

UNCHECK_GEN_QREC: Disables the functionality of generating replication queue records for single and multiple tables (which are separated by comma (,)).

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode=UNCHECK_GEN_QREC;TABLE=Customer" >>
%CODEDIR%\repl_log\batch.log
```

Parameters

TABLE

Name of the mapped table. You can specify a single table name, multiple table names or ALL.

ADD_THREAD: Adds a new replication thread.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param "PATH=C:\Pro2\;DB=All;Mode=ADD_THREAD;THREAD_NUM=10"
>> %CODEDIR%\repl_log\batch.log
```

Parameters

THREAD_NUM

A number for the thread you are adding. The value must be an INTEGER ranging between 1 and 99.

DEL_MAP_TABLE: Deletes a mapped table along with its mapped fields for single and multiple tables (which are separated by comma (,)).

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode=DEL_MAP_TABLE;TABLE=Customer" >> %CODEDIR%\repl_log\batch.log
```

Parameters

`TABLE`

Name of the mapped table. You can specify a single table name, multiple table names or ALL.

`DEL_DBMAP`: Deletes the mapped records of a database.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param "PATH=C:\Pro2\;DB=sports;Mode=DEL_DBMAP" >>
%CODEDIR%\repl_log\batch.log
```

Parameters

`DB`

Name of the database. You can specify a single database name or ALL.

`CHECK_CRC`: Checks whether the CRC matches for mapped tables in the source databases and the local databases in a WAN environment.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param "PATH=C:\Pro2\;DB=All;Mode=CHECK_CRC" >>
%CODEDIR%\repl_log\batch.log
```

Parameters

`DB`

Name of the database. You can specify a single database name or ALL.

`SYNC_REPL`: Synchronizes the replication database table values in a WAN environment.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param "PATH=C:\Pro2\;DB=All;Mode=SYNC_REPL" >>
%CODEDIR%\repl_log\batch.log
```

`CDC Purge`: The CDC Purge menu item can be used to delete the data of change records other than the old data based on user input about the number of days of data that should be deleted and the maximum number of records that should be deleted per batch.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\CDCPurge.p -param
"DB=sports,CDC_PURGE_DAYS=0,MAXPROCESSED=100" > %CODEDIR%\repl_log\CDCPurge.log
```

Processor procedure generator

The processor procedure generator creates a single procedure library that contains an internal procedure for each of the replication tables. These internal procedures encapsulate the replication logic specific to a specific table. The resulting procedure library is run persistently while the replication processor is running. If this procedure is run from the Progress editor, then all databases (source database, schema holder and target database) must be connected.

Additionally, create the following database aliases:

- `SourceDB` for the source database
- `SchemaDB` for the OpenEdge schema holder database.
- `TargetDB` for the target database (which can be an Microsoft SQL Server, Oracle database, or an OpenEdge database).

Pro2 repl databases and table schemas

To facilitate data replication from the source OpenEdge database to the target database, Pro2 uses two light weight databases that contain a series of replication tables. The replication tables manage

Pro2 Repl tables consist of minimal data and indexing, and a single sequence for process control. Replication triggers fire and capture the updated record information to any database replication table as the events take place. A replication table can be in a standalone Repl database or embedded into one of the source OpenEdge databases. To replicate Pro2 uses a replication processor that cycles through replication records periodically, based on user configuration, and replicates the table data directly to the target database.

As an alternate setup, the Repl tables can be embedded directly in the source database.

A ReplQueue is a table that is stored in the replication database. The ReplQueue table stores information on change events that occur on your source database. This information includes the Progress ROWID of the record changed, event date/time, and queue thread. Typically, ReplQueue records represent updates made to the source database that are waiting to be written to the target SQL database.

The Pro2 database and replication database are similar to one another in that they monitor data change events from the ReplQueue, and send those change events to the target database so the that the information can be replicated there.

Table 5: Sequences

Sequence Name	Notes
`NextReplNbr`	Increments the Sequence Number for each New ReplQueue Record

Table 6: Repl Tables

The following are a list of Replication database Tables:

Name	Label
ReplControl	Replication Control
ReplCustAsgn	Custom Assignments
ReplCustDefs	Custom Definitions
ReplCustFlds	Custom Fields
ReplDBXRef	Database Cross Reference
ReplFieldXref	Column Cross-Reference
ReplProperties	Properties
ReplQueue	Replication Queue
ReplTableXRef	Table Cross-Reference

Table 7: ReplControl – Replication control

Field name	Data type	Notes
GroupID	Character	Control Group
CodeID	Character	Group Sub Code
CodeVal1	Character	First Sub Code Filter
CodeVal2	Character	Second Sub Code Filter
CodeVal3	Character	Record Value

Index name	Components	Unique	Primary
idxControl	GroupID	Yes	Yes
	CodeID		
	CodeVal1		
	CodeVal2		

ReplCustAsgn - Custom assignments

No fields or indexes are defined at this time.

Table 8: ReplCustDefs - Custom definitions

Field name	Data type	Notes
SrcDB	Character	Source Database Name
SrcTable	Character	Source Table Name
CustName	Character	Name
CustDefType	Character	Definition Type (Variable, Temp-Table, or Buffer)
CustMisc	Character	If set to CustDefType the Data Type is a Variable. Otherwise, is the name of the table that this temp-table or buffer corresponds to.

Table 9: ReplCustFlds - Custom fields

Field name	Data type	Notes
SrcDB	Character	Source Database Name
SrcTable	Character	Source Table Name
FldName	Character	Name of this field
FldDataType	Character	Data type
FldWidth	Integer	Maximum width of a field. It applies to Decimal and Varchar Fields only
FldDec	Integer	Maximum number of Decimals
FldMand	Logical	Indicates if a field is mandatory field. It is set to No by default.

Table 10: ReplDBXRef - Database cross-reference

Field name	Data type	Notes
SrcDB	Character	Source Database
SchHldr	Character	Schema Holder DB
SchImg	Character	Target Schema Image
TgtType	Character	Target DB Type
TgtConnName	Character	Target DB Connection
TgtPhysName	Character	Target DB Physical Name
GenQRec	Logical	Generate Queue Record

Field name	Data type	Notes
ProcQRec	Logical	Process Queue Record
SrcPhysName	Character	Source DB Physical Name
SchPhysName	Character	Schema Holder DB Physical

Index name	Components	Unique	Primary
idxDBXRef	SrcDB	Yes	Yes
idxDBType	TgtType		

Table 11: ReplFieldXRef - Field cross-reference

Field name	Data type	Notes
SrcDB	Character	Source Database
SrcTable	Character	Source table
SrcField	Character	Source field
SrcDataType	Character	Source data type
SrcOrder	Integer	Field Order
SchField	Character	Schema Field
SchDataType	Character	Schema Data Type
TgtField	Character	Target Field
TgtDataType	Character	Target Data Type
TgtPrec	Integer	Target Precision
TgtScale	Integer	Scale
SrcExtent	Integer	Source Extent
TgtExtent	Integer	Target Extent
OverrideDefs	Logical	Override Precision/Scale Defaults

Index name	Components	Unique	Primary
idxFldXRef	SrcDB	Yes	Yes
	SrcTable		
	SrcField		
idxSrcOrder	SrcDB		
	SrcTable		
	SrcOrder		

Table 12: ReplProperties – Replication properties

Field name	Data type	Notes
PropertyName	Character	Property Name
PropertyValue	Character	Property Value

Index name	Components	Unique	Primary
idxProperties	PropertyName	Yes	Yes

Table 13: ReplQueue - Replication Queue

Field name	Data type	Notes
Sequence	Integer	Sequence Number
EventType	Character	Event Type Single Letter – Create, Write, Delete
SrcDB	Character	Replication Record's Source Database
SrcTable	Character	Replication Record's Source Table
SrcRecord	Character	Source Record ROWID
EventDate	Date	Date Replication Record was Generated
EventTime	Character	Time Replication Record was Generated
SrcTransID	Integer	Source Database Transaction Number
Username	Character	User Id of Transaction
Applied	Logical	Replication Record Processed over to SQL
ApplDate	Date	Date Replication Record was Processed

Field name	Data type	Notes
ApplTime	Character	Time Replication Record was Applied
Audited	Logical	Audited? (Verification)
AudDate	Date	Audited Date of Verification
AudTime	Character	Audit Time of Verification
UserCust	Character	User Custom Data
RawData	Raw	Field to Store RAW information about the Record
QThread	Integer	Replication Queue Processing Thread #
Sequence	Integer	Sequence Number

Index name	Components	Unique	Primary
idxApplied	Applied	No	No
	QThread		
	Sequence		
idxAudited	Audited	No	No
idxSeq	Sequence	Yes	Yes
idxSrcRecord	SrcRecord	No	No
idxSrcTable	SrcTable	No	No

Table 14: ReplTableXRef - Table cross-reference

Field name	Data type	Notes
SrcDB	Character	Source Database
SrcTable	Character	Source table
SchTable	Character	Schema Table
TgtTable	Character	Target Table
GenQRec	Logical	Generate Queue Record
ProcQRec	Logical	Process Queue Record
QThread	Integer	Queue Thread #

Field name	Data type	Notes
UseInDiff	Logical	Include in Differential
TrigInst	Logical	Trigger Installed
MrgdTrig	Logical	Merged Triggers
OrigDTrigProc	Character	Original Delete Trigger
OrigWTrigProc	Character	Original Write Trigger

Index name	Components	Unique	Primary
idxDiff	UseInDiff	No	No
idxTblXRef	SrcDB	Yes	Yes
	SrcTable		
idxThread	QThread	No	No

Pro2 Tables

Name	Label
ReplP_Alerts	Replication Control
ReplCustAsgn	Custom Assignments
ReplCustDefs	Custom Definitions
ReplCustFlds	Custom Fields
ReplDBXRef	Database Cross Reference
ReplFieldXref	Column Cross-Reference
ReplProperties	Properties
ReplQueue	Replication Queue
ReplTableXRef	Table Cross-Reference

Pro2 directory hierarchy

The following is a description of the folders under the Pro2 root installation folder.

Folder	Description
bprepl	Root directory for application programs
bprepl\AppSrv	Used in WAN implementations
bprepl\images	Image files that can be used for shortcut icons
bprepl\misc	Miscellaneous replication files and procedures
bprepl\repl_as_tgt	Used in WAN implementations
bprepl\repl_d	Directory for the Pro2 generated database replication delete trigger procedures
bprepl\repl_export	Used in WAN bulk loads
bprepl\repl_inc	Directory for Pro2 generated assign include files
bprepl\repl_jtrig	Directory for any java triggers
bprepl\repl_log	Default location of log files
bprepl\repl_mgtrig	Directory listing tables that require merged triggers
bprepl\repl_mproc	Directory for the Pro2 generated bulk copy procedures
bprepl\repl_proc	Directory for the Pro2 generated replication library
bprepl\repl_w	Directory for Pro2 generated database replication write trigger procedures
bprepl\repl_tmpl	Directory containing the templates used for various code generation
bprepl\Scripts	Directory containing the .pf files, scripts, and shortcuts to start the Pro2 Admin Tool, replication processors, and bulk load procedures.
bprepl\SQL_inc	Directory for the direct SQL assign include files
bprepl\SQL_mproc	Directory for the direct SQL bulk copy procedures
bprepl\SQL_proc	Directory for the direct SQL replication procedures
custom	Directory to deploy your customized code so that it overrides the existing code in the bprepl folder. **Note:** This folder must maintain the bprepl folder structure and is applicable for both LAN and WAN configurations.
db	Location of schema holder database(s). Also, the initial temporary location for repl database during implementation.

Folder	Description
Installation	Initially empty. Used to save installation site-specific files.
misc	Miscellaneous Pro2 utilities
tmp	Miscellaneous Pro2 temporary files used during implementation

Pro2 program files

Most of the Pro2 source files contain a preprocessor definition for INSTALL_DIR that needs to be set to the directory path, fully qualified in Universal Naming Convention format that contains the bprepl installation directory. This needs to occur before any compile process.

Table 15: Primary programs

Folder	Description
GenReplTrigs.p	Generates the replication triggers. Trigger procedures are created for REPLICATION-WRITE and REPLICATION-DELETE for all mapped tables according to the current database map.
GenReplProc.p	Generates a set of procedures consisting of an individual procedure for each of the tables mapped according to the current database map. Each procedure contains the logic necessary to properly replicate a single record from a single table in a single Source database to a single table in the corresponding target database.
ReplProc.p	Replication Processor – Loads all Replication Processor Libraries into persistent memory and handles the physical replication of data from source to target. The processor program periodically cycles through pending replication records and calls the appropriate procedure from the procedure library. The cycle period is controlled by the current value of the ReplControl sequence in the Replication database. The value is an integer representing the number of seconds to pause between processing cycles.
RunReplProc.p	Initializes the routines for ReplProc.p
bpAdmin.w	The administration and monitoring program for the replication functionality. Replication processing can be monitored from this program. Cycle period setting can be modified, all configuration parameters can be set, certain options can be set, and the GenReplTrigs.p, GenReplProc.p, GenSQLSchema.p and UpdSrcSchema.p programs can be run.
bpadmin.wrx	COM Object support file for bpAdmin.w
ReplAbout.w	Displays version information.
About.txt	Includes the versioning text.

Folder	Description
GenBulkCopy.p	Generates a procedure library consisting of an individual procedure for each of the tables mapped in the current database map. Each procedure contains the logic to mass-copy all records in the appropriate table from the source database over to the target database via the DataServer.
CompReplTrig.p	Compiles the procedure files generated with the GenReplTrigs.p and GenReplProc.p programs.
CompReplProc.p	
UpdSrcSchema.p	Updates the Source database listed in the current database map by creating REPLICATION-WRITE and REPLICATION-DELETE triggers within the meta-schema and then pointing to the appropriate trigger procedure files.
GenSQLSchema.p	Using the current database map, builds a SQL Create Table file in Microsoft SQL format which is used to construct a schema in Microsoft SQL Server that matches the schema of the Progress source database Generates a .sql file consisting of CREATE TABLE and CREATE [UNIQUE] INDEX statements to properly develop an exact copy of the Source schema on a Microsoft SQL Server database. This procedure also adds a “prrowid” field in varchar format that holds the ROWID of the source record.
GenSQLDiff.p	Using the current database map, builds a SQL file in Microsoft SQL format that is used to upgrade the target schema to match the source schema. Generates a .sql file consisting of the necessary SQL commands needed to bring the schema of target database in sync with the schema of source db. Typically used after a schema upgrade to the source db.
LoadFields.p	Used to populate the Mapping screen with Fields, corresponding to the selected Table Map, that are not already mapped for replication.
LoadTables.p	Used to populate the Mapping screen with Tables, corresponding to the selected Database Map, that are not already mapped for replication.
MapAllFields.p	Used to map all fields associated with a Table that has just been auto-mapped.
MapFields.p	Used to cross-reference a single field in the Source table to a single field in the target table.
SetAlias.p	Creates Database Alias Names (SourceDB, SchemaDB, and TargetDB) and/or points them to appropriate Source, Schema and Target Databases according to the current map in use.
repl_tmpl\tmpl_replproc.p	Template File used by GenReplProc.p to create the processor library.
repl_tmpl\tmpl_mreplproc.p	Template File used by GenMassRepl.p to create the bulk copy library.
repl_tmpl\tplt_replTrig.p	Template File used by GenReplTrigs.p to create the Replication Triggers.

Folder	Description
SetAlias.p	Creates Database Alias Names (SourceDB, SchemaDB, and TargetDB) and/or points them to appropriate Source, Schema and Target Databases according to the current map in use.
SQLGenMassRepl.p	Used to generate bulk load procedures for Send-SQL.
SQLGenMSSAssign.p	Used to generate include files for field assignments for Send-SQL procedures.
SQLGenReplProc.p	Used to generate replication procedures for Send-SQL.
SQLReplProc.p	Used to run Send-SQL replication procedures.
SQLRunReplProc.p	Initializes the routines for SQLReplProc.p
purgepro2ent.p	As part of regular maintenance, this program is used to purge the following tables: • Ent_Alerts • Ent_ThreadActDet • Ent_ThreadActivity • Ent_ThreadLag • Ent_ThreadErrors To schedule a data purge, use the DATA_RETENTION_DAYS_ENT property.

Table 16: Secondary programs and files

Program Name	Program Description
ReplLib.i	Contains forward prototype declarations for member functions contained in the replication utility procedure library.
ReplLib.p	A procedure library containing internal procedures and functions used to interface with the tables of the Replication Database and with appropriate configuration and map files.

Table 17: Generated programs

File Name	Function
<ddir>/d<dbName>_<tableName>.p where *ddir* = Delete trigger directory Ex: bprepl\repl_d\dsports_customer.p	One procedure for each replicating table containing the logic to create a replication record for deletion. Each procedure is created in the delete trigger directory specified in the properties table.

File Name	Function
wdir/w *dbName_tableName*.p where *wdir*= Write trigger directory Ex: bprepl\repl_w\wsports_customer.p	One procedure for each replicating table containing the logic to create a replication record for creation or write. Each procedure is created in the write trigger directory specified in the properties table.
*pdir/dbName*_replproc.p *pdir/dbName_tableName*_replProc.p *pdir*/rp_*dbName_tableName*.i where *pdir* = Procedure Library directory Ex: bprepl\repl_proc\rpsports_customer.p	A control procedure containing an individual procedure for each replicating table to handle all basic replication functionality between the source and target databases. These procedures are placed in the processor directory specified in the properties table. There is a replproc.p control program for each database.

Replication procedure library

The replication procedure library (ReplLib.p) contains internal procedures and functions that handle general utility functions, such as reading from and writing to configuration files, map files, and the replication properties and controls tables. It is run persistently and is added to either the session or procedure super stack.

Replication procedure library (ReplLib.p) reference:

- The replication procedure library returns the name of the kill event using RETURN-VALUE.
- It subscribes to the ReplLibPing event which requires an OUTPUT parameter of HANDLE type. It then returns the handle of the persistent instance of the library. And it is used to determine if the procedure is currently running.
- The replication procedure library publishes no events.

Member Functions

FUNCTION	Description	Parameters	Return
formatDate	Returns the input date value as a string in the form YYYY-MM-DD and appends Midnight in the format HH:MM:SS to it.	INPUT DATE – The date to be formatted	CHARACTER – The formatted date
formatLogical	Returns the input Logical variable as either a 1 (True) or a 0 (False) in Integer format.	INPUT LOGICAL – The logical variable to be converted	INTEGER – a Zero for False values or a One (1) for True values

FUNCTION	Description	Parameters	Return
getModDate	Takes the input date and performs various manipulations to convert it into a string such as Weekday, Month dd, yyyy.	INPUT DATE – The date to be converted	CHARACTER – a string interpretation of the date as per Description
getModTime	Takes the input time as an integer and outputs it as a character string in the format HH:MM:SS (24 Hour Time format).	INPUT INTEGER – An integer value representing the elapsed time of the current day, in seconds	CHARACTER – a string interpretation of the time in "HH:MM:SS" format

PROCEDURE	Description	Parameters
LoadConfiguration	Reads a pre-formatted configuration file and places the appropriate sections of that file into the control and properties tables of the Replication Database. This procedure overwrites current settings.	INPUT CHARACTER – a string representing the name of the file from which to load.
LoadMapFile	Reads a pre-formatted map file and places the appropriate sections of that file into the database, table, and field cross-reference tables of the Replication Database. This procedure overwrites current maps with the same cross-reference codes.	INPUT CHARACTER – a string representing the name of the file from which to load.

Replication processor

The replication processor, also known as the thread, is a batch program that periodically cycles through and processes any replication queue records that were not applied to the SQL database. The name of the thread is the timestamp and the number of the thread.

The cycle period is controlled by the sleep interval set by the **Settings** button on the **Dashboard** window of the Pro2 wed interface. This value represents the number of seconds that the processor sleeps between cycles. If the value is set to 0, then the batch session exits and must be restarted to resume processing of replication records.

Replication threads

There can be up to five separate threads of replication processors running. For each thread, a separate replBatch.bat from bprepl\Scripts is run (For example: replBatch.bat, replBatch2.bat, replBatch3.bat corresponding to each thread.) Tables are assigned to specific threads in the Properties window of the Pro2 web interface.

Note: To increase the number of replication threads, contact your administrator to modify your license and get additional threads.

For most implementations, one thread is typically enough to maintain pace with the changes to be replicated. For implementations with very high transaction numbers, tables can be spread across up to five queues to manage performance. Individual tables that have exceedingly high transaction volume or are of very high priority for users of the target database can be assigned their own thread.

Other reasons to have tables in separate threads include running static reports on the target database for a subset of tables. These tables are put in their own thread. When it is required to run the static report on the target database, replication for that thread is stopped. Changes for these tables continue to accumulate. When the report is completed, the replication processor for this thread restarts, catches up with the changes in the queue, and then resumes processing the new changes.

Replication log files

Each replication processor writes to two log files `replproc(YEARMMDD-thread#).log` and `repllog(YEARMMDD-thread#).log`, that are located in the operating system directory specified by the `LOG_DIRECTORY` value in the properties table (by default `bprepl\repl_log` folder). Any errors encountered and general information such as cycle start and stop time and number of records processed are captured here.

These files should be reviewed periodically and require periodic truncation, deletion, or archiving. When the logging level option is set to a setting other than None, the replication processor generates basic log entries, such as when procedure libraries are executed and when the processor queue is started and stopped. Each level builds on the previous level. For example, the Moderate setting logs errors as well as the basic entries generated by the Minimum setting.

Plan for disaster recovery with Pro2

The role of a Replicated SQL database in your business is a major point to consider in planning for Disaster Recovery (D/R) in a Pro2 replication environment. Another point to consider is the size of your source and target databases and the length of time required to restore and to reload the data to the SQL target.

When is it necessary to reset the SQL target database?

Generally, the target SQL database must be loaded whenever the source database or repl database need to be restored from a backup. Bulk loading the data from the source OpenEdge database to the target SQL database can take a long time. This time may range up to days depending upon the size of the database. Your disaster recovery plan should be clear on what situations require a full bulk load of the data and the steps required to do so.

To answer the question of when to reset, it is important to understand the role of `ROWID` in the replication process. The `ROWID` is the unique record identifier in the OpenEdge database. Pro2 uses the `ROWID` to store and identify the corresponding record in the SQL database. Two important facts about how `ROWID`'s are assigned in the OpenEdge database are:

- The order in which records are created and deleted determines their `ROWID`.
- When a record is deleted from the OpenEdge database, its `ROWID` can be used again for a subsequently created record.

Therefore, even if you know which records were created and deleted during the time the source database or repl database were lost and you reapply those changes to a restored version of the database, the `ROWID`'s could be different than the original values stored in the target SQL database and any subsequent changes result in invalid data in the target SQL database.

Why back up the repl database?

The replication database contains valuable information. Understanding the roles of the various tables in the replication database is helpful in determining a backup strategy for the replication database.

Reasons to back up the database are:

- The repl database contains all the mapping information for the replication of the databases, tables and fields between the source and target databases.
- The repl properties file contain configuration settings such as log file location, logical delete tables and specification of procedure templates which are stored in the `ReplProperties` table.
- The `replqueue` table stores information about change events. This information includes the `ROWID` of the record changed, event date and time, and queue thread.

Restrictions with online backups

You must be cautious when you use online backups as part of your disaster recovery solution. Some of the important points to take note of are:

- The backups of the source database and the repl database could be out of sync depending on the timing of the online backups for the repl database and the source database.
- The replication queue may have transactions that were not included in the backup of the source database if the source database backup occurred first.
- The replication queue in the backup of the repl database might be missing committed transactions if the repl database backup occurred first.

Progress recommends that online backups are used only in conjunction with after-imaging.

Pro2 and after-imaging

After-imaging can be enabled and an on the source database concurrently with Pro2 replication. If after-imaging is enabled and running on the source database, then after-imaging does not need to be enabled on the replication database. However, if after-imaging is not enabled on the replication database, the disaster recovery plan must include re-seeding the target SQL database.

Pro2 and OpenEdge Replication

OpenEdge Replication is a Progress product that maintains a replicated copy of the source database as part of a disaster recovery scheme. Pro2 replication can run concurrently with OpenEdge Replication. If the target SQL database and the replication database are part of the disaster recovery solution, then the OpenEdge Replication configuration must be modified to include them.