



Configuration and Administration Guide

Pro2 Replication Suite

OpenEdge®

© 2018 Progress Software Corporation and/or one of 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.

Business Making Progress, Corticon, DataDirect (and design), DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, Deliver More Than Expected, Icenium, Kendo UI, Making Software Work Together, NativeScript, OpenEdge, Powered by Progress, Progress, Progress Software Developers Network, Rollbase, RulesCloud, RulesWorld, SequeLink, Sitefinity (and Design), SpeedScript, Stylus Studio, TeamPulse, Telerik, Telerik (and Design), Test Studio, and WebSpeed are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries. AccelEvent, Analytics360, AppsAlive, AppServer, Arcade, BravePoint, BusinessEdge, DataDirect Spy, DataDirect SupportLink, DevCraft, DigitalFactory, Fiddler, Future Proof, High Performance Integration, JustCode, JustDecompile, JustMock, JustTrace, OpenAccess, ProDataSet, Progress Arcade, Progress Profiles, Progress Results, Progress RFID, Progress Software, ProVision, PSE Pro, SectorAlliance, Sitefinity, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, WebClient, Who Makes Progress, and Xervo 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.

Please refer to the Release Notes applicable to the particular Progress product release for any third-party acknowledgements required to be provided in the documentation associated with the Progress product.

Table of Contents

Table of Contents.....	4
PRO2 Replication Implementation.....	8
Implementation Schedule.....	9
Prerequisites.....	9
Day 1.....	9
Day 2.....	9
Day 3.....	9
Pro2 Server Specification Recommendations.....	10
Operating System	10
CPU.....	10
Memory	10
Network Speed	10
Disk	10
Installation	11
Installing Pro2	11
Upgrading Pro2	13
Uninstalling Pro2.....	15
The Repl Database	15
Stand-alone Repl Database	15
Embedding the Repl Tables.....	16
Repl Database Startup Parameters.....	16
Max Number of Users (-n, maxusers)	16
Max Lock Table Entries (-L, locktableentries).....	16
Max Number of Servers (-Mn, maxservers)	16
Before-Image File Parameters.....	17
Establishing secure database connections	17
Setting up secure connection to a stand-alone database.....	17
Setting up secure connection to a multi-tenant database	17
Second Pass Replication	18
Setting up second pass replication	18

Pro2 Replication Triggers.....	20
Inserting Replication Triggers	20
Removing Replication Triggers.....	21
Schema Trigger Definitions	21
Trigger Procedures	21
Merged Triggers.....	22
Pro2 Disaster Recovery Planning	22
When is it Necessary to Re-Seed the SQL Target Database?	23
Why Back-up the Repl Database?	23
Mapping Information	23
Repl Properties	24
Repl Queue	24
Note of Caution with Online Backups.....	24
Pro2 and After-Imaging	24
Pro2 and OE Replication	24
Configuring Progress to SQL Connectivity.....	25
ODBC Driver Configuration.....	25
Progress Schema Holder Creation.....	28
Testing the DataServer Connection	30
Pro2 Scripts and Parameter Files	31
Parameter Files	31
Shortcuts	32
Scripts	32
Updating Pro2.ini File	33
Pro2 Log Files	33
Log File Maintenance	34

Pro2 Bulk-Load Processor	34
Bulk-Load Procedures	34
Bulk-Load Utility	35
Quit Bulk Load Operation	35
Reset/Clear Bulk-Copy Results	37
Reset Bulk Load Control Records Utility	38
Multi-Table Bulk-Load Criteria Specification	38
For Mapped Bulk-Load Runners Increase Run Flag	38
For Mapped Repl Thread	38
ASCII Bulk-Export/Load	39
Alternative Bulk-Load Options	39
Generating SQL scripts using a Microsoft SQL Server Linked Server	39
Bulk loading data using Pentaho Data Integration	Error! Bookmark not defined.
Pro2 Email Alarms	41
How it works	42
Resetting the email alarm	42
Using a third-party email program	43
Updating Pro2 after Source Schema Updates	43
Generating the Target DB Differential	43
Applying the Update	44
Updating the Progress Image	44
Updating Pro2 Maps	47
Regenerating the Replication Code	48
Setting up Pro2 on UNIX	48
Configuring and Deploying the Pro2 Web Interface	49
Create a replperf database	49
Run the PASOE_WIN_5_x.bat file under Scripts directory	50
Start the PAS instance and login to Pro2Web	50
Setting colors in the Dashboard	51

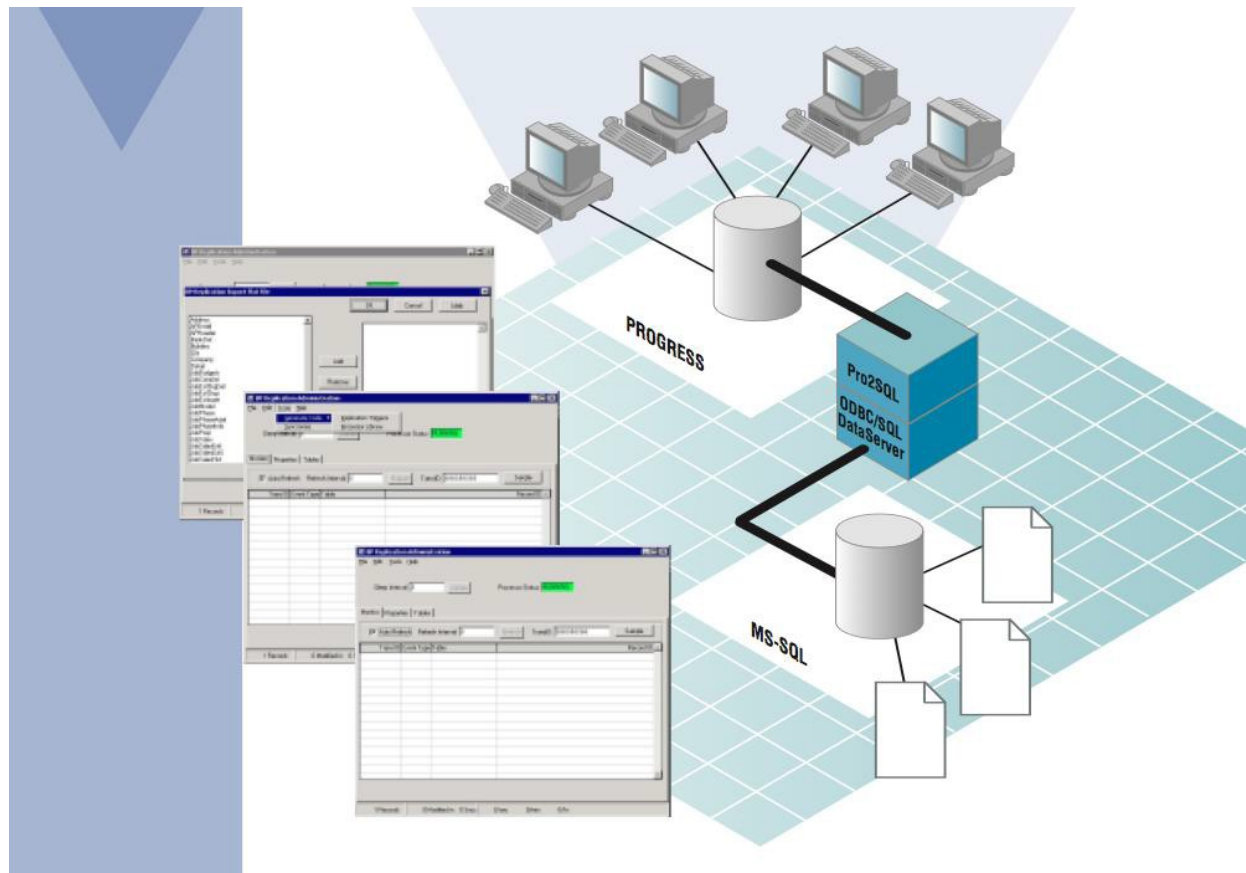
Creating a copy of Pro2	51
Pro2 Replication FAQ - Frequently Asked Questions	51
When do I need to regenerate code?	51
When do I need to update table or field mapping?	51
When do I need to update the schema holder?	52
How do I contact Pro2 Technical Support?	52
APPENDIX A: Repl Tables Schema Objects	53
Sequences	53
Tables	53
ReplControl – Replication Control	53
ReplCustAsgn– Custom Assignments	53
ReplCustDefs – Custom Definitions	53
ReplCustFlds – Custom Fields	54
ReplDBXRef – Database Cross-Reference	54
ReplFieldXRef – Field Cross-Reference	54
ReplProperties – Replication Properties	55
ReplQueue – Replication Queue	55
ReplTableXRef – Table Cross-Reference	56
Appendix B: Pro2 Directory Hierarchy	57
Appendix C: Pro2 Program Files	58
Primary Programs	58
Secondary Programs and Files	59
Generated Programs	60
Appendix D: Replication Procedure Library	61
ReplLib.p	61
Member Functions	61
FUNCTION formatDate	61
FUNCTION formatLogical	62
FUNCTION getModDate	62
FUNCTION getModTime	62
PROCEDURE LoadConfiguration	62
PROCEDURE LoadMapFile	62

PRO2 Replication Implementation

The following diagram describes how this document will refer to each component of the Pro2 Replication configuration:



Note: These components may share physical server machines or, more typically, exist on separate servers. Often the Pro2 server is the same as the SQL server. The Pro2 server must run Windows and can be a virtual machine (VM).



Implementation Schedule

Typically, **Progress Software Corporation** resources install and configure Pro2 Replication. Below is a typical implementation schedule.

Prerequisites

Ensure that the Pro2 Server box and Target Database Server must be configured and functional.

Day 1

1. Finalize the table/fields to be replicated.
2. Install Progress Client/Networking and DataServer on Pro2 server.
3. Build the replication database.
4. Ensure Oracle or MSSQL Server is installed and functional.
5. Identify all PF files and INI files that need modification for 1) connecting to the new replication database and 2) needing PROPATH modifications to find the replication triggers.
6. Identify the placement of the various Pro2 components (i.e. Where do we want the replication database? Where should the QUEUE process reside, etc.)
7. Identify any problem areas, tables too large, indexes too large, etc. (ex: SQL Server is limited to an 8K record size. Are there any issues?)

Day 2

1. Install the Pro2 code.
2. Generate the SQL schema and build the target database.
3. Build the replication map.
4. Generate the replication triggers.
5. Generate the replication procedures.
6. Generate the bulk copy procedures
7. Modify Pro2 icons/shortcuts to accommodate actual installation parameters.
8. Test connectivity to source and target databases.

At this point we need downtime to insert the triggers and modify the PF and INI files to accommodate the Pro2 changes.

Day 3

1. Ensure there are no issues with the PF and INI modifications. (i.e. Are users functioning normally?)
2. Verify if the queue records are being generated as expected.
3. Ensure that the replication process correctly move data from queue to target.
4. Test the bulk copy procedures on a few of the smaller tables.
5. Determine how bulk copy is to be accomplished. (i.e. Are we multi-threading the process? How many threads?)
6. Start bulk copy.

WAN implementations require additional configuration and typically take another day or two.

Pro2 Server Specification Recommendations

The specifications for a Pro2 server are generally the same whether the server is a physical or virtual machine. **Progress Software Corporation** recommends letting the operating system (OS) being loaded on the box determine the specs. The general guidelines are listed below.

Important: These are minimum recommendations. All parameters may be increased to accommodate bigger, faster hardware.

Operating System

- Windows 32-bit or 64-bit
- If this box will contain the SQL Server in addition to the Pro2 product, we would recommend the use of a 64-bit platform with a minimum 8 gig of memory.

CPU

- Minimum 2 processors

Memory

- 64-bit OS - 8 gig of RAM
- 32-bit OS - 4 gig of RAM

Network Speed

- 1 gigabit or better.

Disk

- The physical footprint of Pro2 and the Progress installation is 2.5 Gig.
Note: Pro2 logs can get quite large so we recommend a minimum of 30 Gig to accommodate the logs.
Please note, if this box is to contain the SQL database, the size of the SQL database will be approximately two times the size of the Progress database.

Note: In either a physical or virtual environment it is common practice to spread the I/O. In a physical machine this would mean separating the OS/Pro2 and/or SQL database onto separate physical drives. In a virtual machine the same rules apply where “disks” become multi VMDK files (virtual disk files.) For example, separate the OS and Pro2 install across separate VMDK files.

Installation

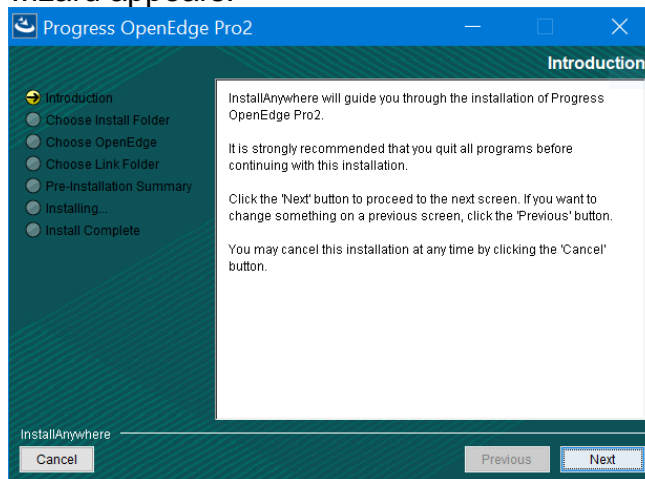
This section describes how to install, upgrade, and uninstall Pro2 in Windows.

Installing Pro2

To install Pro2 in Windows:

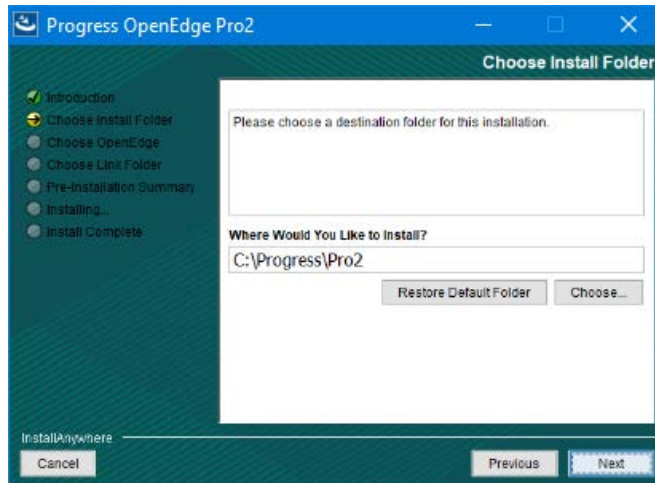
1. Run the Pro2 executable file.

The **Introduction** window of the **Progress OpenEdge Pro2** installation wizard appears.



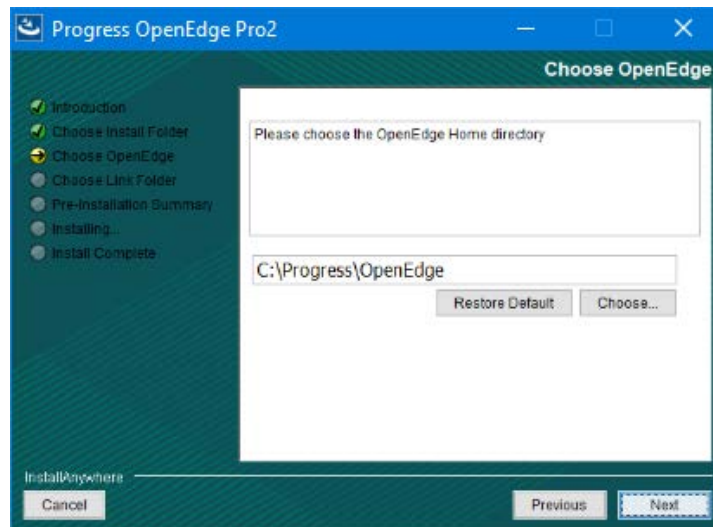
2. Click **Next** to view the **Choose Install Folder** page.
3. Either specify the path or click **Choose** to browse and select the folder where you want to install Pro2.

To select the default installation directory C:\Progress\Pro2, click **Restore Default Folder**.



4. Click **Next** to view the **Choose OpenEdge** page.
5. Either specify the path or click **Choose** to browse and select the OpenEdge home directory.

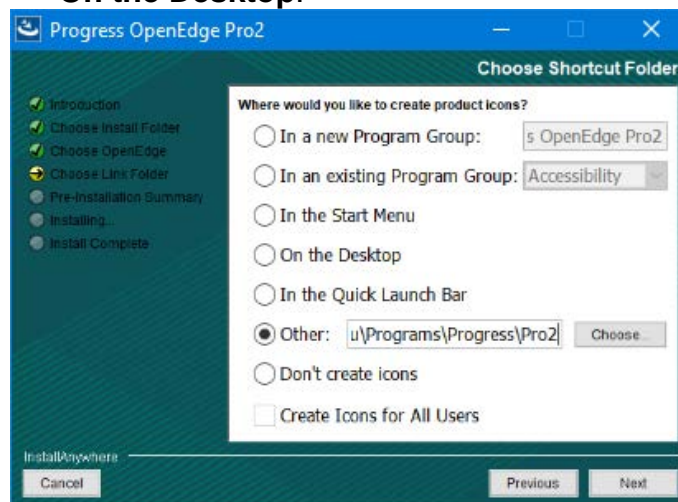
To select the default directory C:\Progress\OpenEdge, click **Restore Default**.



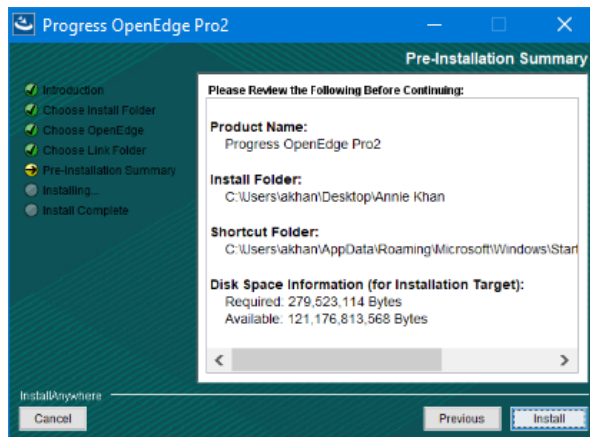
6. Click **Next** to view the **Choose Shortcut Folder** page.
7. Choose one of the following options where you want to create a shortcut icon for Pro2:

- In a new Program Group
- In an existing Program Group
- In the Start Menu
- On the Desktop
- In the Quick Launch Bar
- Other
- Don't create icons

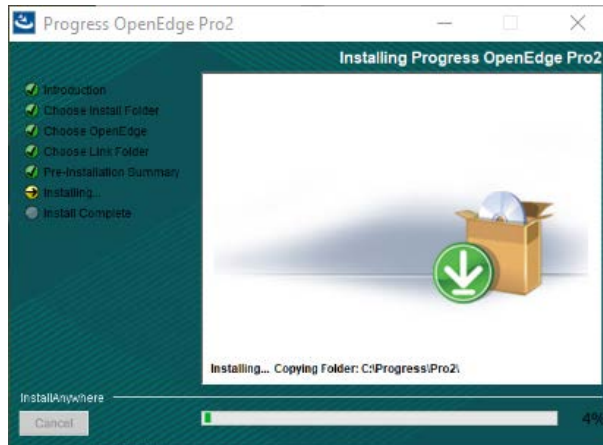
You can select the **Create Icons for All Users** option only when these options are selected: **In a new Program Group**, **In the Start Menu**, and **On the Desktop**.



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



9. Click **Install** to continue with the installation process.



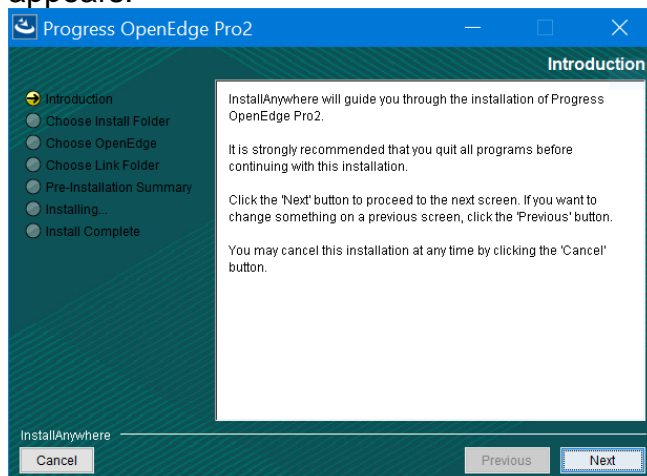
10. When completed, click **Done** to close the installation wizard.

Note: For any post-installation errors, please see the installation log files located in the C:\Progress\Pro2\install\logs folder.

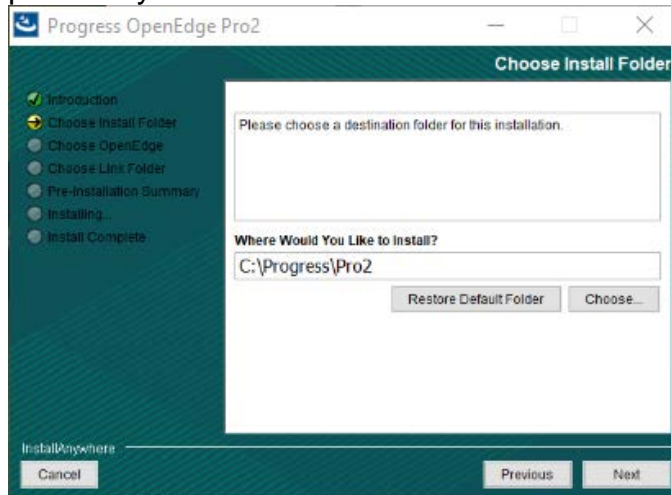
Upgrading Pro2

To upgrade Pro2 in Windows:

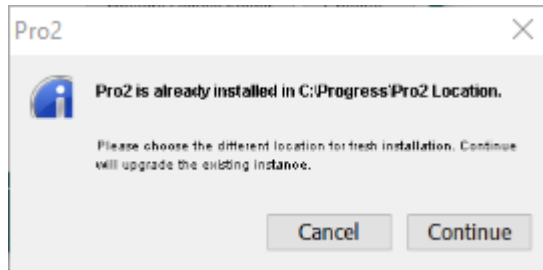
1. Run the Pro2 executable file.
2. The **Introduction** page of the **Progress OpenEdge Pro2** installation wizard appears.



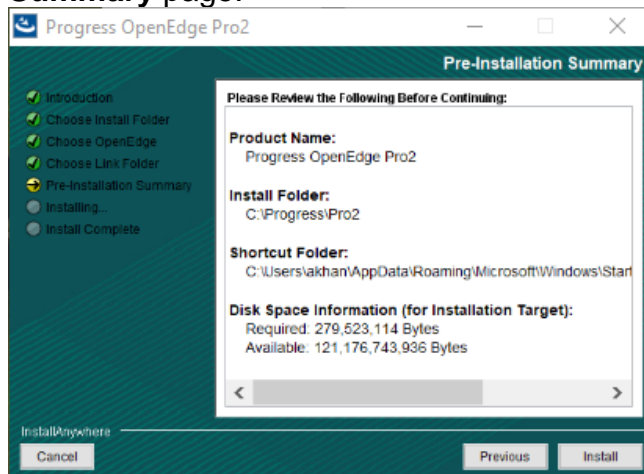
3. Click **Next** to view the **Choose Install Folder** page.
4. Either specify the folder path or click **Choose** to select the folder where you previously installed Pro2.



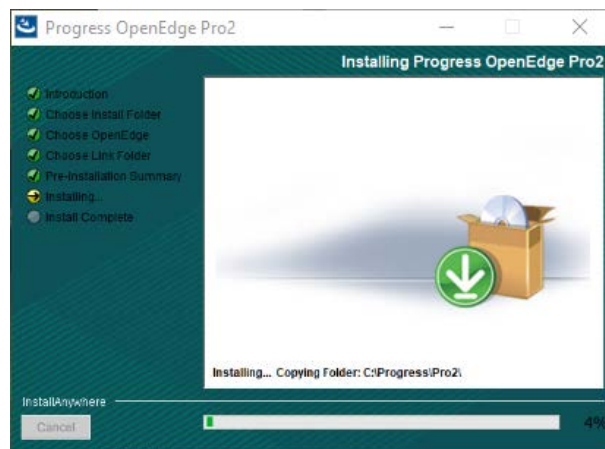
5. Click **Next** and then click **Continue** in the dialog box that appears.



6. Review the options you chose for this upgrade in the **Pre-Installation Summary** page.



7. Click **Install** to continue with the upgrade.



8. When completed, click **Done** to close the installation wizard.

The upgrade creates a **Pro2_backup** folder in the Pro2 installation directory. The backup folder contains your previous Pro2 installation and configuration files.

Uninstalling Pro2

To uninstall Pro2 in Windows, choose one of the following:

- Using the Uninstall utility:
 1. Either run the Uninstall utility from the Pro2 program group or run the **uninstall.exe** file located in the C:\Progress\Pro2\uninstall directory. The **Uninstall Progress OpenEdge Pro2** wizard appears.
 2. Click **Uninstall** to continue with the uninstallation process.
 3. When completed, click **Done** to close the uninstallation wizard.
- Using the Change or remove a program utility:
 1. Open the **Change or remove a program** utility located in Windows Control Panel.
 2. Select **Pro2** from the list of programs and click **Uninstall/Change**.
 3. Click **Yes** to confirm.

The Repl Database

Pro2 Replication defines and uses nine **Progress OpenEdge** database tables. These nine tables can be embedded in one of the source databases or they can be configured as a stand-alone Progress database. See Appendix A: Repl Tables Schema Objects for a description of the **repl** tables.

Stand-alone Repl Database

Standard Progress database utilities are used to create the repl database. The structure file (.st file) and schema definition file (.df file) used to create the repl database can be found in the **root Pro2 install folder**.

All end-user configuration files, parameter (.pf) files, and scripts will need to be modified to include a connection to the repl database wherever there is a

connection made to the source database. The `ubroker.properties` file will need to be modified for Progress AppServers that update the source database to add a connection to the repl db.

Another point to consider is **conmgr.properties** modifications, start/shutdown scripts and backups. A stand-alone repl database will require standard database administration. This includes `conmgr.properties` modifications, start-up/shutdown scripts and backups. See the **Pro2 Disaster Recovery Planning** section for more information.

Embedding the Repl Tables

Embedding the repl tables into the source database can simplify implementation. If the repl tables are embedded, no additional database connection will be required.

One major reason customers choose to keep repl as a stand-alone database is to simplify schema updates made by the application provider to the source database. Pro2 works the same whether the repl tables are embedded or in a separate stand-alone database.

Repl Database Startup Parameters

Max Number of Users (-n, maxusers)

All end-users must connect to the repl database as well as to the source database. For this reason, the max number of users (the **-n** parameter when starting the database server with the `proserve` command or **maxusers** when using `dbman`) specified for the repl database should not be lower than that of the source database. If this parameter is not set high enough, end-users may not be able to run the application.

Max Lock Table Entries (-L, locktableentries)

Each update to the source database, which requires an Exclusive-Lock on the source record, results in a record being written to the replqueue in the repl database which also requires an Exclusive-Lock. Thus, the max number of lock table entries (the **-L** when starting the database server with the `proserve` command or **lock table entries** setting in the **conmgr.properties** file when using `dbman`) specified for the repl database should not be lower than that of the source database. If this parameter is not set high enough, end-users may get errors while trying to make updates and those updates will fail.

Max Number of Servers (-Mn, maxservers)

If there are a high number of remote users connecting to the source database, the default of five servers to handle all remote connections to the repl database may not be sufficient. The max number of servers for the repl database (the **-Mn** when starting the database server with the `proserve` command or **maxservers** setting in the **conmgr.properties** file when using `dbman`) should not be lower than that of the source database. If this parameter is not set high enough, end-users connected as remote clients may not be able to run the application.

Before-Image File Parameters

For optimal performance, the repl database before-image (BI) file should be truncated at initial implementation to set the BI block size (-biblocksize) and the BI cluster size (-bi). Ex: **proutil repl -C truncate bi -biblocksize 16 -bi 16384.**

Establishing secure database connections

Pro2 enables you to validate and regulate access to your database, standalone or multi-tenant for security purposes.

Setting up secure connection to a stand-alone database

To establish a secure connection to a stand-alone database:

1. Specify the username and password in the <database>.pf file if you are connecting to the database for the first time.
2. Open the Pro2 Administration tool, click **Tools > Generate Encrypted Password**, and specify the credentials that you want to use and click **OK**. This generates a **db_login.txt** file in the Pro2 installation directory.
3. In the Pro2 installation directory, edit the **predefs.i** file and perform the following:
 - a. Uncomment &GLOBAL-DEFINE USERIDDB yoursourcedb, and replace 'yoursourcedb' with the name of the source database to which you want to connect.
 - b. Uncomment {bprepl/dbsecure.i}.
4. Remove the credentials specified in step 1.

The Pro2 Admin tool is now set to connect to your stand-alone database securely using your credentials.

Setting up secure connection to a multi-tenant database

With Pro2, you can configure a secure connection to your multi-tenant database. Each tenant in a multi-tenant database must have its own installation of Pro2, a schema holder, and an ODBC connection. However, for all the tenants, the logical name of the ODBC connection can be same, and the schema holder name can be same as the Pro2 installation folders are different for each tenant.

To establish a secure connection to a multi-tenant database:

1. Embed Repl database into your source database by performing the following:
 - a. Shutdown the source database.
 - b. In the repladd.st file, add the Repl database structure to the source database by updating the command prostrct add <sourcedb_name> repladd.st.
 - c. Load repl.df to the new database.
 - d. Start the source database (proserve <sourcedb_name>)
2. Create a separate folder for each tenant, for example Pro2_tenant1, Pro2_tenant2, etc., and install Pro2 for each tenant.
3. In the Pro2installdir\bprepl\scripts folder, right-click **Pro2 Administrator** shortcut and select **Properties**.

4. In the **Shortcut** tab, modify the **Start in** field to point to its corresponding tenant.
5. Similarly, modify the **Start in** field of **Pro2 Editor** and **RunBulkLoader** files to point to their corresponding tenants.
6. Edit the Pro2_env file and update the paths to point to the corresponding tenant.
7. Create <database>.pf files for each tenant, for example Tenant1DB.pf, Tenant2DB.pf, etc., and edit them to point to their corresponding Pro2 instances.
8. Specify the username and password in the <database>.pf files if you are connecting to the database for the first time.
9. Open the Pro2 Administration tool, click **Tools > Generate Encrypted Password**, and specify the credentials that you want to use and click **OK**. This generates a **db_login.txt** file in the Pro2 installation directory.
10. In the Pro2 installation directory, edit the **predefs.i** file and perform the following:
 - a. Uncomment &GLOBAL-DEFINE USERIDDB yoursourcedb, and replace yoursourcedb with the name of the source database to which you want to connect.
 - b. Uncomment {bprepl/dbsecure.i}.
11. Remove the credentials specified in step 8.

The Pro2 Admin tool is now set to connect to your multi-tenant database securely using your credentials.

Second Pass Replication

Second Pass Replication helps users to perform audit data warehouse tasks. It helps track all data changes to a table during a time interval for any or all fields. Second Pass replication creates a copy of the source table in the **Second Pass** database and inserts a record into this table every time there is a change in the source table (for Update, Delete, Insert). A new record is created in the target (or staging) table everytime there is a change in the table irrespective 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, **AUDIT_PASS**, was added to the **predefs.i** file. This property is to support 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. Audit Pass Replication is supported for both LAN and WAN setups. The example provided in this section uses a WAN setup.

After Second Pass Replication is set up, the **Admin Tool** title is suffixed with **AUDIT** in parenthesis and all the elements on the Admin tool becomes read-only. However, the user can still run the code generation menu items in the WAN menu.

Setting up second pass replication

Follow the steps in this section to set up Second Pass Replication:
Pro2 Config and Admin Guide

To set up Second Pass Replication:

1. Create a copy of the Pro2 setup in directory and rename it to append it with SecondPass. For example, if your setup is in the folder P2_WAN, copy paste that folder and rename it to **P2_WAN_SecondPass**.
2. Create a desktop shortcut to the scripts folder in the **SecondPass** setup and rename it with to append it with SecondPass. Scripts folder is in <setup directory>/ P2_WAN_SecondPass /bprepl.
3. Right click the **Pro2 – Administrator** tool from <setup directory>/ P2_WAN_SecondPass /bprepl/scripts folder and select **Properties**.
4. On the Shortcut tab, edit the **Start in** field to reflect the P2_WAN_SecondPass folder.
5. Right click the **Pro2 – Editor** tool from <setup directory>/ P2_WAN_SecondPass /bprepl/scripts folder and select **Properties**.
6. On the Shortcut tab, edit the **Start in** field to reflect the P2_WAN_SecondPass folder.
7. Edit the **Pro2_env.bat** file to set the value of the Pro2 setup location to the Second Pass directory.
8. Delete the contents of the <setup directory>/ P2_WAN_SecondPass/DB folder.
9. Open the replProc.pf file from the <setup directory>/ P2_WAN_SecondPass/bprepl/scripts folder and modify the Replication Database section to point to the Pro2 setup.

Note: This is not the Second Pass directory but the Pro2 setup directory.

10. Save the contents of the **replProc.pf** file.
11. From the <setup directory>/ P2_WAN_SecondPass /bprepl/scripts folder, edit the sports2000.pf file and modify the Source Databases section to point to the DB in the Pro2 setup.

Note: This is not the Second Pass directory but the Pro2 setup directory.

12. Save the contents of the sports2000.pf file.
13. Create a new target database with the SecondPass suffix in the name. For example, if your target database is sports2000, name the new database as **sports2000_WAN_SecondPass**.
14. Create a new **ODBC Data Source name** for the new Second Pass target database.
15. Build a schema holder with the name **sports2ksh**.
16. Create a new ODBC Data Source, S2K_WAN_SecondPass and build a schema holder with the name Sports2KSH.
 - a. Using Proenv, run the command <setup directory>:P2_WAN_SecondPass\ldb>prodb sports 2ksh empty.
 - b. Run the command, <setup directory>:P2_WAN_SecondPass\ldb>prowin sports2KSH -1

- c. Setup a schema holder from the Database Administration tool and point it to the new ODBC Data Source name
17. From the `<setup directory>/P2_WAN_SecondPass/bprepl/scripts` folder, edit the **sports2000.pf** file to modify the Schema Images section to point to your ODBC data source, `s2k_wan_SecondPass`, in this case. Save the file.
18. From `<setup directory>/P2_WAN_SecondPass/bprepl/repl_tmpl`, rename the process library template name in Second Pass setup directory.
For a **WAN** setup, rename the file `tmpl_replproc_WAN_4audit.p` to `tmpl_asTgtDb.p`.
For a **LAN** setup, rename the file `tmpl_replproc_4audit.p` to `tmpl_replproc.p`.
19. Rename the file **tmpl_replproc_WAN_4audit.p** to **tmpl_asTgtDb.p**.
20. From the `<setup directory>/P2_WAN_SecondPass`, open the **predefs.i** file for the instance that you want to set up Second Pass replication for and change the value for `AUDIT_PASS` to YES.
21. Open the Pro2 Admin tool from the Audit setup to see that all fields are read-only.
22. Generate the process libraries.

Note: Queue Disposition on the Monitor tab should be set to Mark as Applied and not Delete Record.

Pro2 Replication Triggers

Once the replication triggers are inserted into the source database schema, Pro2 Replication is “on”. If configured correctly, end-users will not notice that the triggers are in place. Otherwise two major issues could occur:

1. The client executable cannot locate the trigger procedure. In this case, an error such as “./bprepl/wr<table>.t not found” will be displayed and the update will fail.
2. Since the Pro2 trigger procedures connect to the repl database, if the client is not connected to repl, the following error will occur and the update will fail: “Database repl not connected.”

Once the triggers are inserted into the source side database for all intents and purposes the repl database (or the source database with the repl tables if embedded) becomes another production level database. This means if the repl database is down, the application will not function properly until one of the two following conditions are met: The repl database is restarted or the triggers are removed from the source side database(s).

Inserting Replication Triggers

Exclusive access to the schema is required to insert triggers. If the database is in multi- user mode, there should be no users connected or at least no active transactions. Preferably, the database should be shut down and connected in single-user mode to ensure exclusive access to the schema.

The **replTrigInsert.p** procedure is used to insert the triggers. The procedure is run from the Progress procedure editor connected to both the source database and the repl database (or to the source database with the embedded repl tables). The procedure first needs to be modified to specify the source database logical name in the “vDB” parameter.

If there are multiple source databases, replTrigInsert.p needs to be run against each source database one at a time.

Removing Replication Triggers

As is the case when inserting triggers, exclusive access to the schema is required when removing triggers. The repl database does **not** need to be connected to remove triggers. The **replTrigDel.p** procedure is run against the source database to remove the triggers.

Note: This does not delete the trigger procedures. Once the trigger definitions are removed, the trigger procedures will not be executed even if they remain on the system.

Schema Trigger Definitions

Below is output from the Progress Trigger Report for the Customer table in the sports database after Pro2 triggers have been inserted. To run this report, from the Procedure Editor, run Tools → Data Administration.

From the Data Administration window, select **Database → Reports → Triggers**.

<u>Table/Field Name</u>	<u>Event</u>	<u>CRC</u>	<u>Flags</u>	<u>Procedure</u>
Customer	CREATE	no		C:\dlc101c\sports\crcust.p
	DELETE	no		C:\dlc101c\sports\delcust.p
	RP-DEL	no		.\bprepl\repl_d\dsports_Custom
	RP-WRI	no		.\bprepl\repl_w\wsports_Custom
	WRITE	no		C:\dlc101c\sports\wrcust.p

In the trigger report, the event type “**RP-DEL**” stands for REPLICATION-DELETE and “**RP-WRI**” for REPLICATION-WRITE, which includes create events as well as update events. The procedure specified is run when this type of event occurs.

Schema trigger definitions can also be displayed by the **replTrigDisp.p** procedure (to screen) and the **replTrigDisp2.p** procedure (to file).

Trigger Procedures

Care must be taken in placing the trigger procedures on the database server and on any application server boxes from which end users run the application. The Pro2 trigger procedures must be found by the client executable in a location

specified in the **PROPATH**. If there are other triggers defined for the database, the replication trigger location can be in the same directory as the other triggers since that must be already included in the **PROPATH**.

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 two sub-directories: **repl_d** (delete triggers) and **repl_w** (write triggers) in a location already in the **PROPATH**, otherwise the **PROPATH** will need to be modified to include the location of the **bprepl** directory.

Each time a client executable attempts to update a record in the source database, if a replication trigger is defined in the schema for that table, the client will attempt to run the specified trigger procedure. In cases where the user is connected as a local "direct connect" client (whether it be an end-user or an AppServer servicing a remote client), the trigger procedures must be located on the database server. In cases where the client is a remote client connected to the database via client/server, the trigger procedures will need to be on the client system.

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

Merged Triggers

If pre-existing *replication* triggers (**REPLICATION-DELETE** or **REPLICATION-WRITE**) exist in the source database, the Pro2 replication triggers will need to be merged or integrated with the original triggers. This can be done 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 that there are two types of triggers in Progress, primary triggers and replication triggers which are executed after the primary triggers. **CREATE**, **WRITE**, and **DELETE** are primary triggers and are not impacted by Pro2 replication. These types of triggers do not need to be integrated with Pro2 triggers; only replication triggers (**REPLICATION-DELETE** or **REPLICATION-WRITE**) need to be merged with the Pro2 triggers.

Pro2 Disaster Recovery Planning

Note: Ideally Pro2 is not designed for disaster recovery.

The major point to consider in planning for Disaster Recovery (D/R) in a Pro2 replication environment is the role of the replicated SQL target database to your business. 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.

Pro2 Disaster Recovery Planning should include responses to the questions: How
Pro2 Config and Admin Guide

would it impact our business if we lost the:

- Source database
- Repl database (assuming repl tables are not embedded in a source database and are in a standalone “repl” database)
- SQL target database
- Pro2 Server
- SQL Server
- Database Server

When is it Necessary to Re-Seed the SQL Target Database?

In general, whenever the source database or repl database need to be restored from backup, the target SQL database should be re-loaded. Bulk loading the data from the source Progress database to the target SQL database can take a long time, days or perhaps even weeks for very large databases. Your disaster recovery plan should be clear on what situations will require a full re-bulkload of the data (also known as “re-seed”) and the steps required to do so.

To answer the question of when to re-seed, it is important to understand the role of the Progress ROWID in the Pro2 replication scheme. The ROWID is the unique record identifier in the Progress 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 Progress database are:

- The order in which records are created and deleted determines their ROWID.
- When a record is deleted from the Progress database, its ROWID can be re-used for a subsequently created record.

Thus, even if you know exactly what records were created and deleted during the time the source database or repl database were lost and re-apply those changes to a restored version of the database, the ROWID's could be completely different than the original values stored in the target SQL database and any subsequent changes will result in invalid data in the target SQL database.

Why Back-up the Repl Database?

Understanding the roles of the various tables in the repl database will be helpful in determining a backup strategy for the repl database.

Mapping Information

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 from the **File** menu in the Pro2 Admin Tool.

Repl Properties

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 **File** menu in the Pro2 Admin Tool.

Repl Queue

Information on change events is stored in the ReplQueue table. 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.

Note of Caution with Online Backups

Use caution when using online backups as part of your Disaster Recovery solution. Depending on the timing of the online backups for the repl database and the source database, the backups of the source database and the repl database could be out-of-sync. The ReplQueue may have transactions that were not included in the backup of the source database if the source database backup occurred first, or the ReplQueue in the backup of the repl database might be missing committed transactions if the repl database backup occurred first. Progress Software Corporation recommends that online backups are used only in conjunction with After-imaging.

Refer to **Progress Software's Open Edge Database Administration** documentation for information on Progress backup utilities.

Pro2 and After-Imaging

After-imaging can be enabled and running on the source database concurrently with Pro2 Replication. If After-imaging is enabled and running on the source database, it does not need to be enabled on the repl database. However, if after-imaging is not enabled on the repl database, the Disaster Recovery plan must include re-seeding the target SQL database.

Refer to **Progress Software's Open Edge After-Imaging** documentation for information on how to enable after-imaging.

Pro2 and OE Replication

Open Edge Replication is a Progress product that maintains a replicated copy of the source database as part of a Disaster Recovery scheme. Pro2 Replication can be running concurrently with Open Edge Replication. If the target SQL database and the repl database will be part of the D/R solution, the OE Replication configuration will need to be modified to include them.

Refer to **Progress Software's Open Edge Replication** documentation for more information.

Configuring Progress to SQL Connectivity

To create and configure the Progress DataServer for MSSQL or Oracle required by the replication system, follows the steps detailed in this section. For more information on installing and configuring data servers from Progress, see:

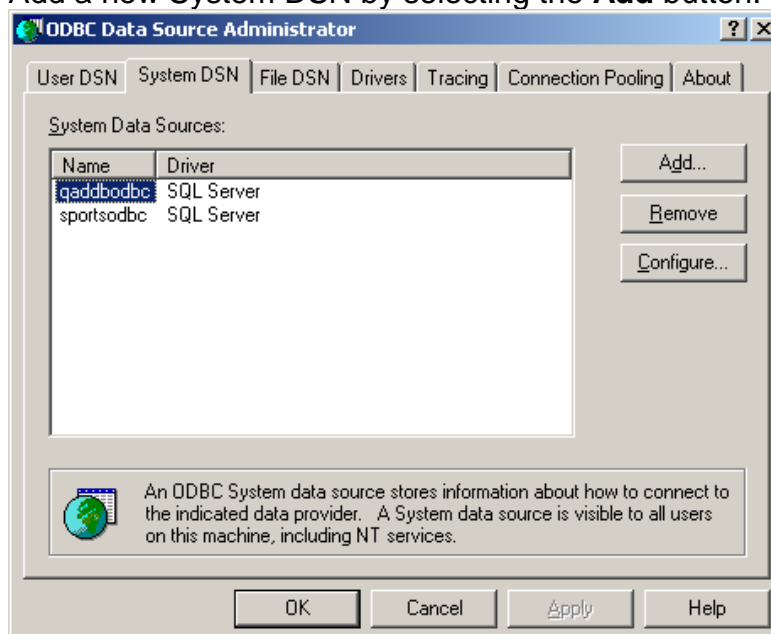
- *Progress OpenEdge Data Management: DataServer for Oracle*
- *Progress OpenEdge Data Management: DataServer for Microsoft SQL Server*

ODBC Driver Configuration

Prior to configuring the Progress DataServer, ODBC connectivity must be setup to allow communication between Progress and foreign database.

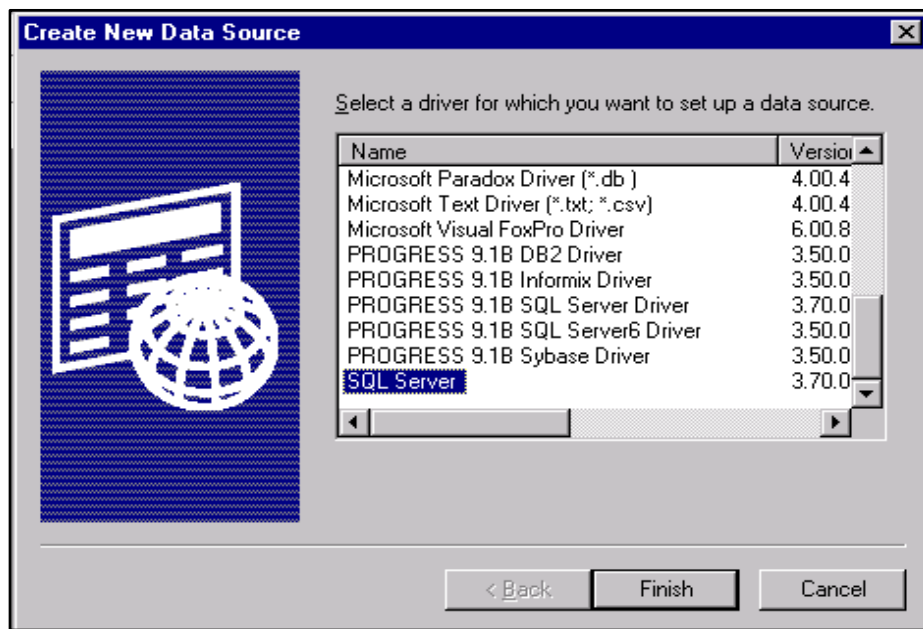
1. Using the Windows Data Sources tool, add a new System DSN
 - a. On 32-bit Windows: Start menu → Settings → Control Panel → Administrative Tools → Data Sources (ODBC) → System DNS Tab
 - b. On a Windows 64-bit system, since the Progress client is available only in 32-bit run the following:
C:\Windows\SysWOW64\odbcad32.exe or
%SYSTEMROOT%\SysWOW64\odbcad32.exe

2. Add a new System DSN by selecting the **Add** button.

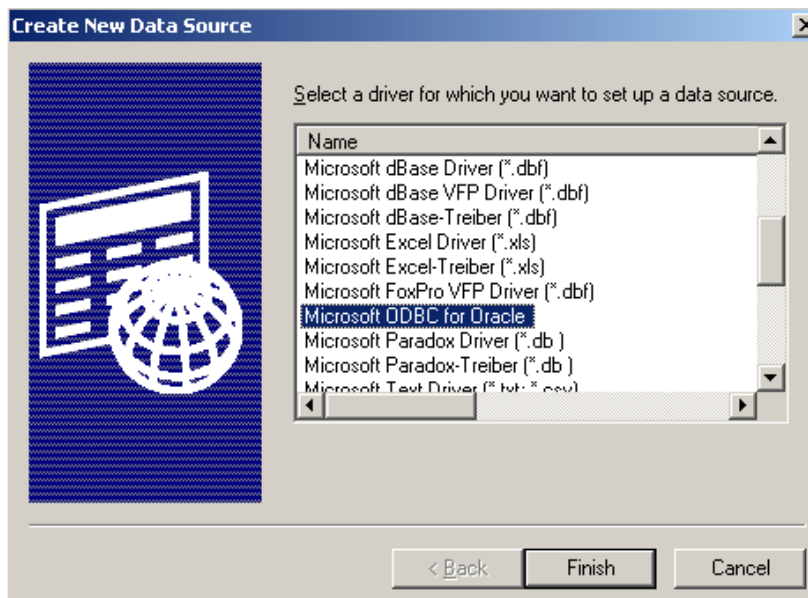


ODBC Data Source Administrator

3. For MSSQL Server, choose the **SQL Server** driver; for Oracle, select the **Microsoft ODBC for Oracle** driver, and click **Finish**.

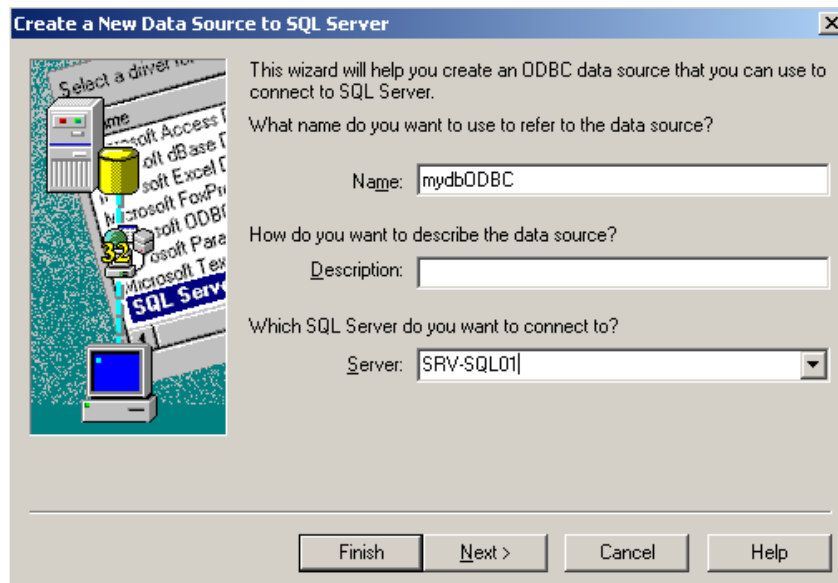


Create New Data Source Dialog for MSSQL Server



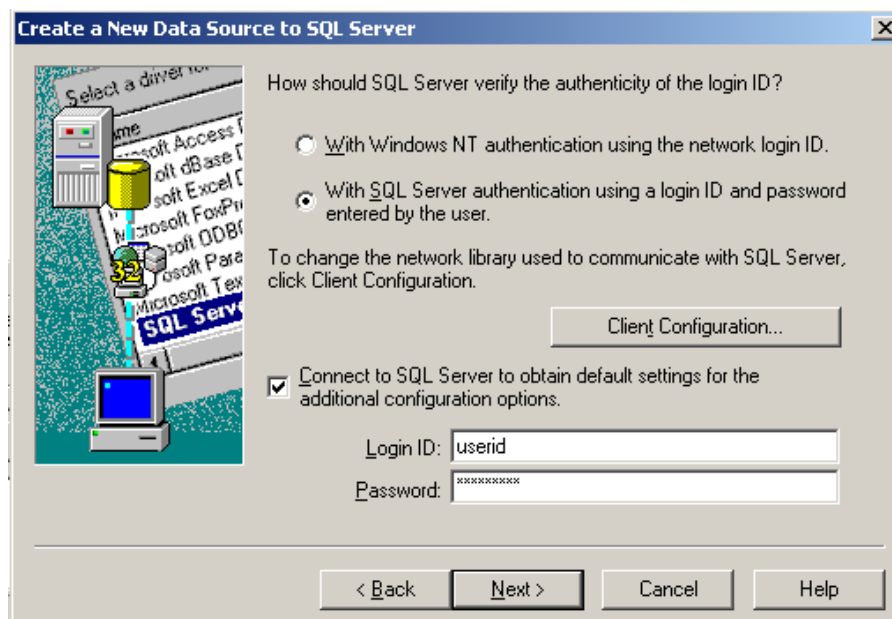
Create New Data Source Dialog for Oracle

- Specify the name (for Pro2 typically <logical-db-name>ODBC) for your data source and the server where it resides, and click **Next**.



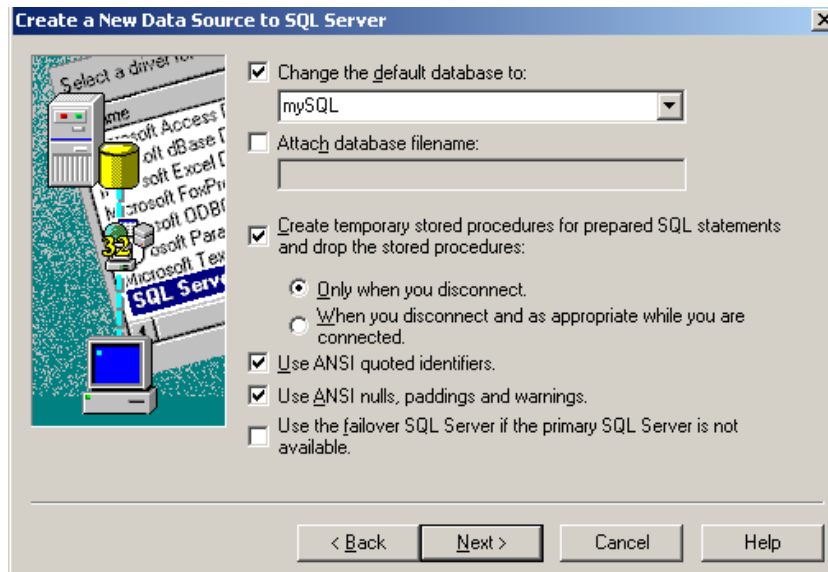
Create New Data Source Dialog (2)

- Specify login information. If using Select SQL authentication, provide the Login ID and Password for the MS SQL database, and click **Next**.



Microsoft SQL Server DSN Configuration Dialog

- From the **Change the default database to** drop-down, select the appropriate database, and click **Next**.



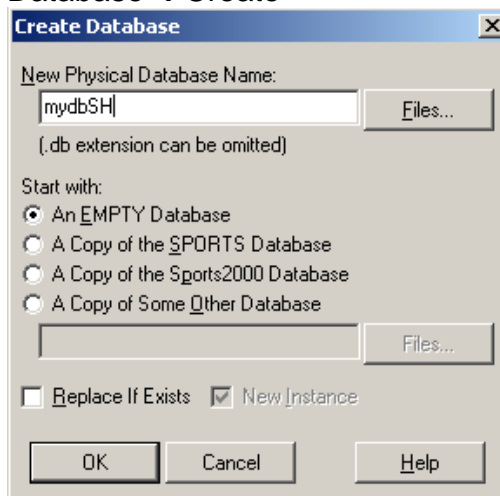
Microsoft SQL Server DSN Configuration Dialog

7. Accept the defaults on the next dialog box and click **Next**.
8. The next dialog box will have a button for testing the connection. When selected, a message should return indicating success.
9. Exit the Data Source administration tool.

Progress Schema Holder Creation

An empty Progress database must be created to act as the schema holder for the MSS DataServer. The schema holder is the layer that will allow the Progress 4GL to interact with the MS SQL Server database as though it were a native Progress database.

1. From the Progress Data Administration tool, select the following from the menu.
 - a. Database → Create



Progress Connect Database Dialog

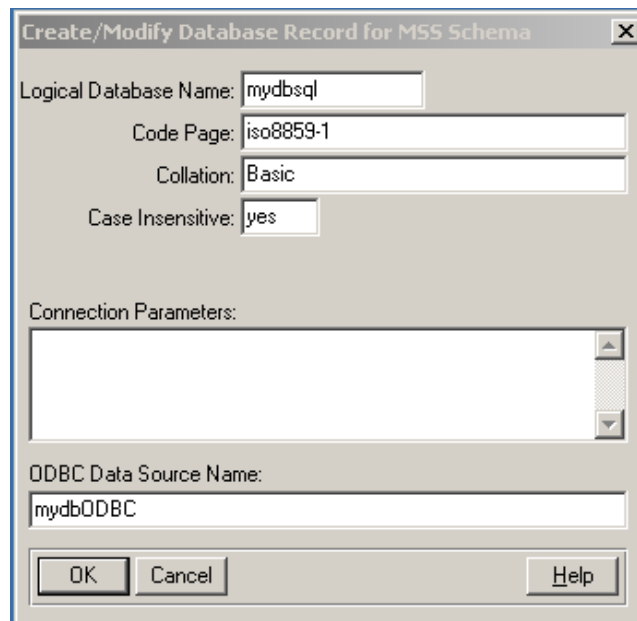
2. Specify a physical database name (mydbSH) and choose the Empty Database to start with, and click **OK**.



The 'Connect Database' dialog box has a title bar with a close button. It contains four input fields: 'Physical Name' with 'mydbSH', 'Logical Name' with 'mydbSH', 'Database Type' with 'PROGRESS', and an empty 'Options >>' field. There are four buttons: 'Browse...' next to Physical Name, 'OK' next to Logical Name, 'Cancel' next to Database Type, and 'Help' next to Options >>.

Progress Connect Database Dialog

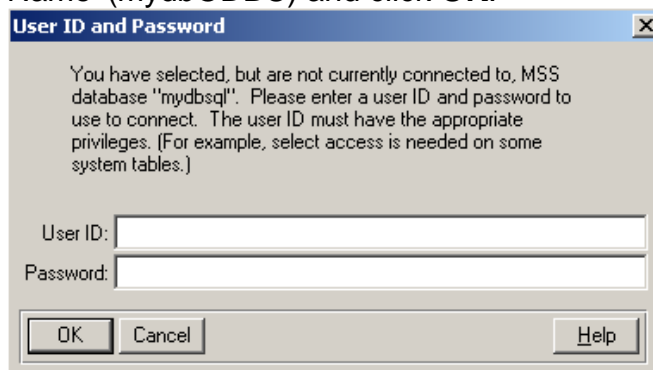
3. If the new database was successfully created the Connect Database dialog box will appear. Click **OK**.
4. From the Database Administration tool, select the following from the menu:
 - a. DataServer → MS SQL Server Utilities → Create DataServer Schema



The 'Create/Modify Database Record for MSS Schema' dialog box has a title bar with a close button. It contains five input fields: 'Logical Database Name' with 'mydbsql', 'Code Page' with 'iso8859-1', 'Collation' with 'Basic', 'Case Insensitive' with 'yes', and 'ODBC Data Source Name' with 'mydbODBC'. There is a 'Connection Parameters' section with a text area. At the bottom are three buttons: 'OK', 'Cancel', and 'Help'.

Progress Create/Modify Database Record Dialog

5. Enter the Logical Database Name (mydbsql) and ODBC Source Name (mydbODBC) and click **OK**.



The 'User ID and Password' dialog box has a title bar with a close button. It contains a message: 'You have selected, but are not currently connected to, MSS database "mydbsql". Please enter a user ID and password to use to connect. The user ID must have the appropriate privileges. (For example, select access is needed on some system tables.)'. Below the message are two input fields: 'User ID' and 'Password'. At the bottom are three buttons: 'OK', 'Cancel', and 'Help'.

Progress Login Dialog

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

Progress Pre-Selection Criteria Dialog box

7. Enter criteria for **Object Name**, **Object Owner** (typically, but not always “DBO”), and **Qualifier**.
8. Select the **Default to OpenEdge DATETIME** option.
9. 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. Now click **OK**.
10. In the next window, select the table names required for replication or choose **Select Some** and enter **Table** for the Object Type to select all tables. Click **OK**. The schema pull will commence and could take 30 minutes or longer for a database with a large number of tables.

Testing the DataServer Connection

From the Progress editor, the following command should connect to the DataServer and a target MSS database via the ODBC data source:

```
CONNECT -db mydbsh
        -db mydbodbc
        -dt MSS
        -ld mydbSQL
        -U <SQL_Login_Name>
        -P <SQL_password>
        -Dsrv binding,0
        -Dsrv TXN_ISOLATION,1.
```

From the Progress editor the following command should connect to the DataServer and a target Oracle database via the ODBC data source:

```
CONNECT -db mydbsh
        -db mydbodbc
        -dt ORACLE
        -ld mydbSQL
        -U <SQL_Login_Name>
        -P <SQL_password>
```

```
-Dsrv binding,0  
-Dsrv TXN_ISOLATION,1.
```

Pro2 Scripts and Parameter Files

Scripts and shortcuts used to launch the Pro2 replication processes are located in the **bprepl\Scripts** folder. The parameter files that specify connection information to the source, target and schema holder databases are also located in this folder.

When maintaining multiple implementations of Pro2 on the same Pro2 server (Ex: having both a production and a test instance such as C:\Pro2Prod and C:\Pro2Test), it is important that the scripts, shortcuts and parameter files be updated for each environment to reflect the correct information for that environment.

Parameter Files

Parameter files are used by the Progress executable (**prowin32** or **_progres** (**prowin** for newer 64 bit versions)) to specify startup parameters and database connection information.

replproc.pf - this is the main parameter file (.pf) that specifies the repl database connection info (-H, -S) and any subsequent .pf files that will be used.

Below is an example of a replProc.pf file where **sports** is the source database.

```
-T tmp  
-mmax 8192  
-TB 31  
-TM 32  
-s 100000  
-h 20  
-yy 1950  
-yr4def  
  
# Replication Database  
-db c:\pro2demo\db\repl -ld repl  
  
# Connect to source and target databases  
-pf c:\pro2demo\bprepl\scripts\sports.pf
```

<DB-Name>.pf - Each source database should have it's own .pf file in the Scripts folder.

<DB-Name> is the name of the source database as specified in the DB Map tab in the Pro2 Admin Tool. (See the **Pro2 User's Guide** for more info.) This .pf file will specify the source database connection info, the schema holder connection info, and the target database connection info (ODBC Data Source).

Below is an example **sports.pf** file.

```
# Example PF File for Database Connections for Pro2
# Actual .pf file should be named after the logical name
# for the Source database connection.

# Source Databases go here.
-db c:\pro2demo\db\mysports -ld sports

# Schema Holder Databases go here.
# If not built yet then just comment out all lines.
-db C:\pro2demo\db\sportssh -ld sportssh -RO

# Target Schema Images go here.
-db sportsodbc      #ODBC Data Source
-ld sportssql      #Logical db name in schema holder
-dt MSS            #MSS for SQL Server, Oracle for Oracle
-Dsrv QT_CACHE_SIZE,30000
-Dsrv TXN_ISOLATION,1
-Dsrv BINDING,0
-Dsrv AUTOCOMMIT,1
-Dsrv PRGRS_PREPCACHE,100
-Dsrv MSS_PRESERVE_CURS,1
-Dsrv PRGRS_LOCK_ERROR,08501
```

Shortcuts

- **Pro2 Administration Tool Shortcut:** Used to start an instance of the Pro2 Admin Tool. The shortcut properties are set to specify the “Start in” folder and replproc.pf for the current instance of Pro2. Note that all references use bprepl/ in the name so set “Start-in” to Pro2 install root folder (ex: D:\Pro2).
- **Pro2 Editor Shortcut:** Used to start an instance of the Progress Procedure Editor connected to the source, target and schema holder databases. The shortcut properties are set to specify the “Start in” folder and replproc.pf for the current instance of Pro2

Scripts

Below is a description of some of the major scripts in the bprepl\Scripts folder.

- **Pro2Env.bat** – Called by other Pro2 scripts to set the environment variables that specify the location of the Progress OpenEdge and Pro2 installs.
- **replBatch.bat** – Kicks off replication processor instance for main queue thread. Typically configured to run via Windows Task Manager. Replbatch2.bat through replbatch5.bat kick off replication processor for threads two through 5.

- **replmbatch.bat** – run by the Bulk Load Utility bulk-load individual tables.

Updating Pro2.ini File

The Pro2 root installation directory provides a Pro2.ini file that implements Pro2 replication.

You must update the OpenEdge installation paths (DLC location) in the following sections of the Pro2.ini file:

```
[Startup]
V6Display=no
;ImmediateDisplay=yes
;MultitaskingInterval=100
DefaultFont=MS Sans Serif, size=8
DefaultFixedFont=Courier New, size=8
DLC=C:\117\dlc
Use-3D-Size=Yes
PROPATH=custom,.
```

```
SysCheckmark=251
DLC=C:\117\dlc
PROPATH=custom,.,C:\117\dlc\tty,
```

```
; This section defines the Java environment.
[JAVA]
JDKHOME=C:\117\dlc\jdk
JREHOME=C:\117\dlc\jre
JVMEXE=java
```

Note: Updating the Pro2.ini file is mandatory and is applicable to all new and previous installations.

Pro2 Log Files

Log files are written to the bprepl\repl_log folder by default or to the folder specified by the LOG_FILE_DIRECTORY property.

The following table lists the various log file settings:

Property	Description	Default Value
ADMIN_LOG_FILE	Logs changes made to replication queue thread settings via Pro2 Admin Tool.	Admin.log
LOG_FILE_DIRECTORY	Location of Pro2 log files	bprepl\repl_log
LOG_FILE_SUFFIX	Extension used on log file names	.log
OUTPUT_BASENAME	Replication processor log file name. Queue thread number and log file suffix are appended to the basename.	replproc

REPL_LOG_BASENAME	Name of file for logging replication of records during each replication cycle. Date, queue thread number, and log file suffix are appended to	repllog
-------------------	---	---------

Log File Maintenance

Pro2 log files can grow quite large, especially if the logging level is set to verbose. Log files that are from previous days can be deleted once replication has switched to a new log file. (Freeware utility **delage** is included in bprepl\Scripts folder and can be used for this purpose if desired.)

Any replprocYYYYMMDD-(thread#).log file can be removed.

Any old repllogYYYYMMDD-(thread#).log can be removed. Do not remove the most current unless replication is stopped.

Pro2 Bulk-Load Processor

The Bulk Load Processor is a collection of special replication programs that perform replication for an entire table. This is used to do an initial sync of the source table to the target table. A bulk-load procedure basically 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. It can also be re-run at any time to re-sync an individual table (with the exceptions of deletes on the target; if source deleted rows exist in the target these will need to be removed manually or, to always be certain this does not happen, you can truncate the data from the target table before performing a bulk-load).

Bulk-Load Procedures

The Bulk-Load procedures can be generated after table mapping has been done from the Admin Utility Tools menu: [Tools]->[Generate code] ->[Bulk-Copy Processor]. The bulk-load procedures are generated in the directory specified by the MASS_LOAD_DIRECTORY property (bprepl/repl_mproc by default).

The template file specified by the MASS_LOAD_TEMPLATE property is used in the bulk-load procedure generation. By default, this procedure is tmpl_mreplproc_restart_auto-push.p. Bulk-load procedures generated using this template will pick up where they left off if they are stopped and restarted. In addition, if a source record is locked and cannot be copied, the procedures will add the record to the replqueue to be written by the replication processor and the bulk-load procedure will move on to the next record.

The Bulk_Max_Cache property in the tmpl_mreplproc_restart-auto-push template stores the last processed row-ids 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 Admin Tool.

During the bulk-load process, the Oracle_Bulk_Transaction_Count property (Oracle only) commits multiple records at a time per transaction. This property defaults to 1000. You can add this property and then modify its value using the **Properties** tab in Pro2 Admin tool.

When setting this property for LAN configuration, ensure that the MASS_LOAD_TEMPLATE property must use the `tmpl_mreplproc_oracle.p` procedure. For WAN, 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 LOBs in or exclude them from replication and bulkload. The property is set to YES by default, which transfers LOB data to the target database. However, to exclude LOBs from replication and bulkload, set this property to NO. This is applicable to both LAN and WAN environments.

All bulk_load templates have ElapsedTime property which indicates the total time taken for the bulk-load process to complete. 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 as well as at the end of the loading process. The bulk-load procedures write to individual log files in the folder specified by the LOG_DIRECTORY property ("bprepl\repl_log" folder).

The Bulk-Load procedures can be run individually from the Procedure Editor or in a group by the `mr<source-database>.p` procedure which runs each one in turn. Multiple bulk-loads can also be launched at once from the Bulk-Load utility.

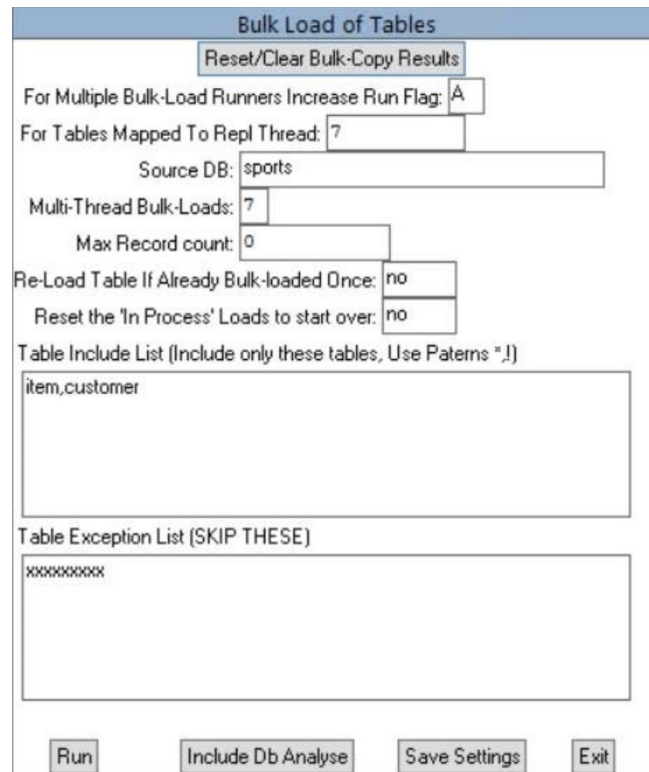
Bulk-Load Utility

The Bulk-Load utility can be run from the Admin Utility Tools menu: [Tools]->[Run Bulk Loads]. It can also be run from the RunBulkLoader shortcut in the `bprepl\Scripts` folder. Note: The Bulk-Load Utility always runs the standard bulk copy procedures in `bprepl\repl_mproc`. It does not run the Direct SQL bulk load procedures in `bprepl\SQL_mproc`. The SQL_mproc procedures must be run from a Pro2 Editor session.

Quit Bulk Load Operation

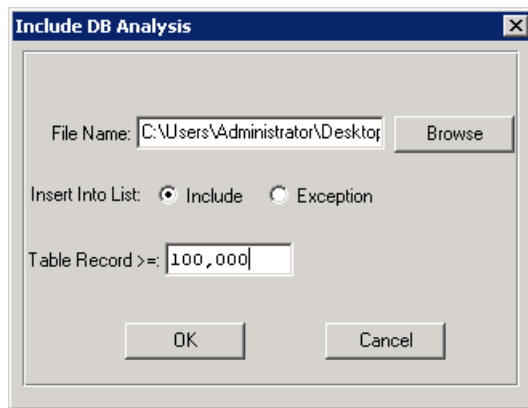
The **Quit Bulk Load** option allows you to **Bulk-load** a specific number of records by specifying the number of records within the table. Once the specified record count are met, the Bulk-load process will auto abort itself. For example, to begin with, you may want to Bulk-load only 100 records from a table which has over 100,000 records. Once 100 records are bulk loaded, the Bulk-load process will end.

You can use this property by manual intervention from the **Pro2 admin** tool. In the **Pro2 admin** tool, click **Tools** tab and Select **Run Bulk Loads**. You will be directed to the **Bulk Load of Tables** window. You can specify the number of records to be Bulk loaded by specifying the number required in the **Max Record Count** section.



Note that customers can choose to bulk load only a few select tables based on the number of records they contain. For example, they may want to bulk load only those tables that have over 100,000 records. Or, alternatively, they may want to bulk load only those tables that have less than 100,000 records. This can be accomplished as follows:

1. Perform a database or table analysis using the DBANALYS or TABANALYS command in **proutil**. Depending on the command that is being run, one or more output files will be generated, including a **<dbname>.tab** file which contains the table analysis.
2. In the Bulk Load Utility, click **Include Db Analyse**. This opens the Include DB Analysis dialog box.



3. In the **Include DB Analysis** dialog box:
 - a. Specify the **<dbname>.tab** output file that contains the table analysis in the **File Name** field.
 - b. Enter the number of records by which you need to filter the tables in the **Table record >=** field.
 - c. In the **Insert Into List** field, choose whether to include the tables that pass the filter (**Include**), or exclude them (**Exception**) during the bulk load execution. If you choose **Include**, the tables that pass the filter will be displayed in the **Tables Included List** in the Bulk Load Utility. If you choose **Exception**, they will be displayed in the **Table Exception List**.
 - d. When you are done, click **OK**.

After specifying the criteria, select [Run]. A dialog box will display the number of tables to be loaded and the number of bulk-load threads that will be spawned. If this information does not look correct, answer [No] and adjust the criteria for the desired tables.

Note: If you enter ALL in the Source DB field, Bulk Load will be run for tables in all the mapped databases. To run Bulk Load for specific databases, enter the names of the Databases in the Source DB field separated by commas.

Otherwise, select “Yes” to start the bulk-copy processes. The individual bulk-load procedures that correspond to tables that meet the criteria entered will be run sequentially (spread amongst the number of bulk-load threads specified) via the **replmBatch.bat** script in the **bprepl\Scripts** folder.

The screen will be updated to indicate which table(s) are in process of bulk-loading. Log files are written to the folder specified by the LOG_DIRECTORY property (by default the “bprepl\repl_log” folder).

Note: If you are performing the Bulk Load process more than once, use the [Reset/Clear Bulk-Copy Results] before running Bulk Load.

Reset/Clear Bulk-Copy Results

The Reset/Clear Bulk-Copy Results button on the Bulk Load utility clears the details captured in the Repl Control table from the previous bulk load.

Reset Bulk Load Control Records Utility

This menu item enables the user to clear all the bulk load control records. To clear all bulk load control records, select Reset Bulk Load Control Records from the Tools menu and in the next popup window, select the databases that you want to include for this action and click **OK**. [Tools]->[Reset Bulk Load Control Records]

Multi-Table Bulk-Load Criteria Specification

The Bulk-Load utility automatically skips over tables that have been flagged to not generate queue records (genqrec = FALSE) or to not process queue records (procqrec = FALSE). All other criteria are user-selected.

For Mapped Bulk-Load Runners Increase Run Flag

If more than one instance of a Bulk-Load Utility is run concurrently, each instance must be given a separate flag (typically letters A through Z) to distinguish temp files created and used by each instance.

Each instance of the Bulk-Load Utility can have up to five bulk-load threads. (See discussion of threads below.) The number of bulk-load utility instances and associated threads is limited by the Progress license on the Pro2 box. For example, two Bulk-Load runners set for five threads each equals a total of twelve Progress license sessions running

For Mapped Repl Thread

Tables mapped to the replication thread entered will be selected; zero (0) specifies all threads.

- **Start At Table:** Indicates what table to start from. Format: {dbname}.{tablename} Example "sports.item" will start at the "item" table of the "sports" database, skipping over tables with names alphabetically before "item". A period (.) indicates to start from the beginning.
- **Multi-Thread Bulk-Loads:** The number bulk-load processes or "threads" to run at once. The maximum is five bulk-load threads per Bulk-Load launcher.
Note: Bulk-load threads are distinct from replQueue threads.
- **Re-Load Table if Already Bulk-loaded Once:** Pro2 track of the tables that have been bulk- loaded stores and stores information in the replControl table. It uses this data to restart a load if it is stopped. It also will skip loading a table that is already completely loaded. If you set this flag to yes it will delete any "COMPLETED" control records so the table will re-load from the beginning.
- **Table Include List:** A comma separated list of tables to include. Patterns can be specified using "*" as a wild card. Example: "ca*,da*,e*,z*". This will bulk-copy all tables starting with ca, da, e, and z.
- **Table Exception List:** A comma separated list of tables to exclude or skip. Patterns can be specified using "*" as a wild card. Example: "ca*,da*,e*,z*". This will bulk-copy all tables except those starting with ca,

da, e, and z.

ASCII Bulk-Export/Load

For WAN implementations, the ASCII bulk-export/import was designed to sync replication tables from Source to Target. For very large tables (typically over fifty million rows), the ASCII Bulk-Export/Load utility can be used to decrease the time of the bulk load.

The process uses an ASCII dump export process on the Source server. This process creates the dump procedures and the SQL load procedures and then dumps the data to ASCII flat files. Once this is finished the dumped ASCII data files and the generated SQL load procedures are transferred to the Target DB server. The SQL load procedures are then run from the SQL query-editor. See the Pro2 WAN documentation for more info.

Alternative Bulk-Load Option

Pro2 provides the following alternative for bulk loading:

- **Generating SQL scripts using a Microsoft SQL Server Linked Server:**
This option is applicable only when your target database is a SQL Server database. You can generate SQL script files containing INSERT INTO statements that you can run on the SQL Server database.

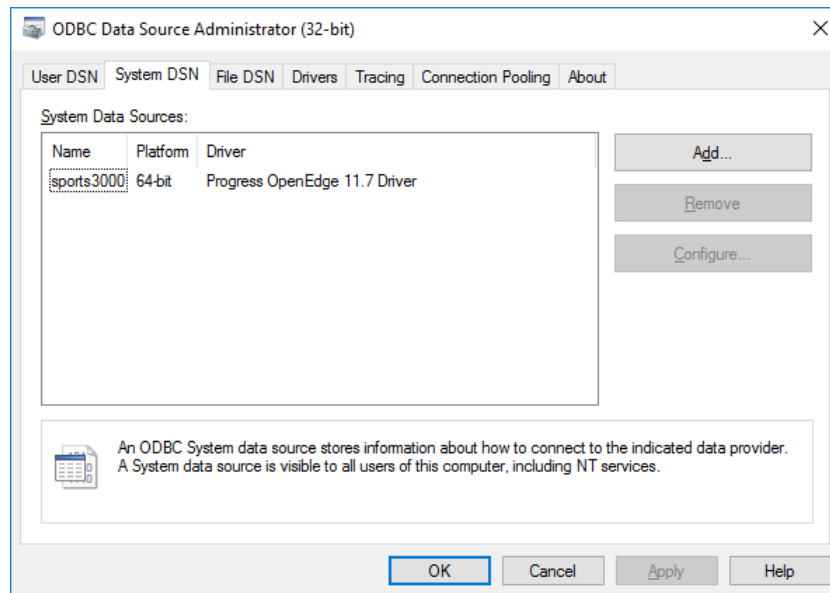
Generating SQL scripts using a Microsoft SQL Server Linked Server

If the target database is a Microsoft SQL Server database, you can generate SQL script files containing INSERT INTO statements to bulk load data from the source OpenEdge database. This is a faster alternative to running bulk procedures. To do this, perform the following steps:

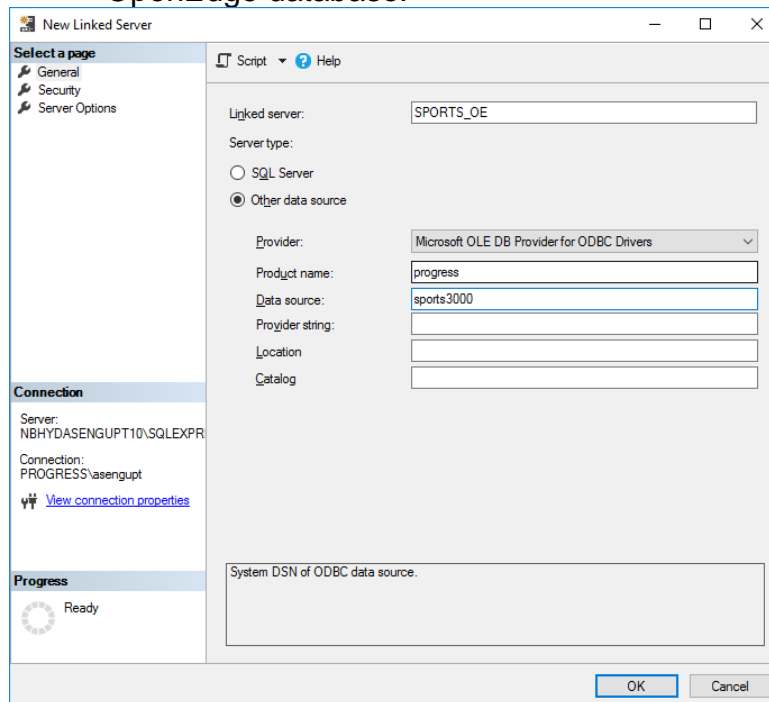
1. In the source OpenEdge database, create a user with DBA permissions. For example, you could run the following commands in Proenv:

```
SQLEXP -db <dbname> -H <host> -S <port>
CREATE USER '<username>', '<password>';
GRANT DBA to '<username>';
COMMIT;
```

2. Create an ODBC system data source for the OpenEdge database using the **ODBC Data Source Administrator** utility. Remember to define the same username and password that you specified while creating the DBA user in Proenv.



3. In the target Microsoft SQL Server database, create a **Linked Server** with the following configuration:
 - General properties:
 - **Provider:** Microsoft OLE DB Provider for ODBC Drivers
 - **Product name:** progress
 - **Data source:** <System data source name that you specified in the ODBC Data Source Administrator utility>
 - Security properties:
 - Choose **Be made using this security context** and specify the username and password that you created for the OpenEdge database.



4. Using the Pro2 Admin Tool, generate the SQL script files as follows:

- a. Select **Tools > Generate Code > Linked Server SQL**.
- b. In the dialog box that opens, enter the name of the **Linked Server** that you created and the **OpenEdge ODBC data source name**. You can also choose to include Pro2 fields in the resulting script files if required. When you are done, click **Submit**.

- c. Specify a location to save the SQL files. One SQL file will be generated for each table in the source OpenEdge database.
Note: You also have the option to generate the SQL scripts by running the **linked_SQL.bat** file located in **bprepl/scripts**. To do this, modify the following parameters in the file before running it:

```

INC_PRO2_FIELDS=<YES_OR_NO>,
LINK_SERVER=<linked_server_name>,
DSN_NAME=<OpenEdge ODBC Data Source Name>

```

5. Finally, execute the SQL files on the target SQL Server database.

Pro2 Email Alarms

Note: The Heartbeat functionality (with email alarm) is applicable for both replication and CDC admin threads.

You can configure Pro2 to send email alarms that are triggered when a replication thread stalls or stops. To detect that a replication thread is stalling, Pro2 checks the timestamp of the thread's most recent replication activity (called its heartbeat) against the time-period specified in a property called HEARTBEAT_CHECK_MINUTES that you can configure in the Pro2 Admin Tool. If, for example, the HEARTBEAT_CHECK_MINUTES is set to 15, and the last recorded activity of the thread was more than 15 minutes from the time of checking, the email alarm is triggered.

To set this up, you need to perform the following steps:

1. Configure the HEARTBEAT_CHECK_MINUTES property in the Pro2 Admin Tool.
2. Edit the **predefs.i** file located in the Pro2 root directory and define the

following email properties:

- FROM-EMAIL-ADDR—the email address of the sender of the email alert.
 - TO-EMAIL-ADDR—the email address of the recipient of the email alert.
 - COMPANY-MAIL-SERVER—the SMTP mail server of the sender of the email alert.
3. Define username/password authentication in a property called SMTP AUTH in the Pro2 Admin Tool. The username and password should be separated by a comma.
 4. Create a new task in Windows **Task Scheduler** to periodically run a batch file called **statusCheck.bat**.

How it works

The process begins with the **statusCheck.bat** script, which executes periodically based on the schedule configured in the Windows task scheduler.

The statusCheck.bat file does only one thing—it runs a procedure called **statusCheck.p**, that is located in **<PRO2_root_directory>/bprepl**. The statusCheck procedure compares the timestamp of the replication thread's most recent activity (the heartbeat) with the value specified in the HEARTBEAT_CHECK_MINUTES property.

If the difference between the timestamp and the system time (at the time of checking) is greater than the value specified in the HEARTBEAT_CHECK_MINUTES property, AND if an email alert has not already been sent on the current date, it forms the subject and body of the email alarm and calls another procedure named **smtpmail.p**.

The smtpmail procedure opens an SMTP connection using the details specified in the **predefs.i** file and sends the email using the subject and body defined by the checkStatus procedure.

The body of the email contains a message that is similar to the following string:

```
It has been approximately 3 minutes over the allotted time of 10 minutes
(Property: HEARTBEAT_CHECK_MINUTES) since a heartbeat was recorded. The last
heartbeat was recorded at 11/09/2017 15:11:04.243
```

If required, you can customize the subject and body of the email by modifying the vSubject and vBody parameters in the checkStatus.p procedure file.

Resetting the email alarm

StatusCheck will send only one email alarm per day to prevent overload of email messages. Once the email has been received and replication restarted, you can reset the alarm so that alerts are sent if replication goes down again on that same day. To do this, select Tools->Reset Alarm in the Pro2 Admin Tool or set the value of the ALERT_1_PER_DAY property to NO.

Using a third-party email program

If you want to use a third-party command-line email program like Blat, perform the following steps:

1. Set the property `USE_3RD_PARTY_EMAIL` to Yes in the Pro2 Admin Tool.
2. Edit **mail.bat** located in `bprepl/scripts` and modify the batch script to send the email. For example for Blat, you would use the following statement:

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

Note that the argument values (%1, %2, etc) 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 email program. For example, %1 denotes the email recipient address, %2 denotes the subject of the email, and so on. The `mail.bat` file contains remarks which explain the purpose of each the variables.

Updating Pro2 after Source Schema Updates

Whenever a Progress schema change is made, and the modification impacts tables currently involved in replication, the changes must be applied to both the replication target database and its related Progress Schema Holder image. The Pro2 Administration Utility contains a menu option that will generate a script to be applied to the target replication database. Once the script is applied, any alterations must then be updated in the schema holder database.

*Refer to **Pro2 WAN implementation** guide for additional instructions.*

Generating the Target DB Differential

The first step in updating Pro2 is generating the script that will update the replication target database. You can perform this step whether replication is running or not. Most of the time, however, replication will be off when you do this. To generate the differential, follow these steps:

1. Open the Pro2 Administration Utility.
2. On the menu, select **Tools → Generate Code → Target Differential**.
3. The next dialog will ask you which database to use as a differential basis.
4. Select the appropriate database and click **OK**. You will then get another dialog box to confirm the database selected and the name of the output file. Click **OK** again. You should get a successful message once the process is complete.
5. The target DB differential scripts will be generated to the directory `bprepl/repl_target` or to the directory you have described as the **TARGET_DIRECTORY** in the **Properties** tab.

The `TARGET_DIRECTORY` potentially will contain up to three database scripts:

- `dbname-DiffAleter.sql` — Contains any ALTER statements to modify EXISTING database tables. This would be any new/changes to fields for

- tables that previously existed.
- dbname-Create.sql — Contains any CREATE TABLE statements to add new tables than exist in the Progress db but do not exist in the replication target database.
 - dbname-Drop.sql — Contains any DROP TABLE statements for tables that exists in the replication target database but do not exist in the Progress database.

NOTE: Read this script carefully. If you added tables to the replication target database that do not exist in the Progress db this script may contain a DROP statement for these tables. Simply ignore this script or remove the database only tables as you see fit.

After you have created your differential scripts, you must review them carefully. You can use any text editor to do this. Essentially, the purpose of this step is to ensure that whatever changes are made to Progress are reflected in the SQL differential script.

Applying the Update

Once you have reviewed your generated Target Differential, it is time to apply it to the replication target database. You will be creating and executing a new query as follows:

1. Shut down replication, if you have not already done so (schema updates require exclusive access to the schema).
2. Open your database querying system of choice.
3. Select the database you are updating and open a new query.
4. Copy and paste the script you created and reviewed into the empty query window.
5. Execute the query.
6. Your database should be updated at this point.

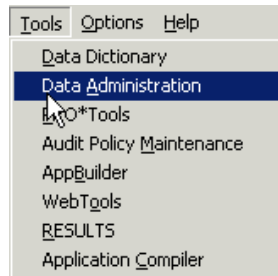
The scripts should not take very long to implement. The longest part of the execution may be the index rebuild (especially if new indexed fields were added to the table).

Updating the Progress Image

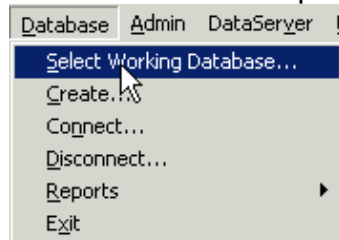
Now that your replication target database accurately reflects the changes made to your Progress database, you will need to update the Progress Schema Holder to account for the modifications. To update your Progress schema image, follow these steps:

1. Close any instances of the Pro2 Administration Utility, as the schema update will require exclusive use of the database.
2. Launch a PROENV cmd session - Start → Programs → OpenEdge → Proenv
3. Cd to the Pro2\DB folder – (Different for each pro2 installation.)
4. Do dir *.db to get the name of the Schema holder - (Different for each pro2

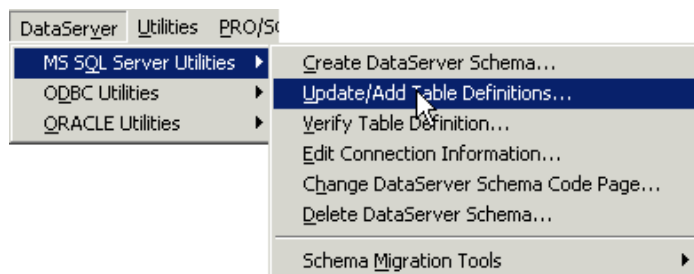
- installation.)
- The schema holder will be something like {sourcedbname}SH.db
 - Do `Prowin32 dbname -1 -rx`
 - On the menu, select **Tools** → **Data Administration**.



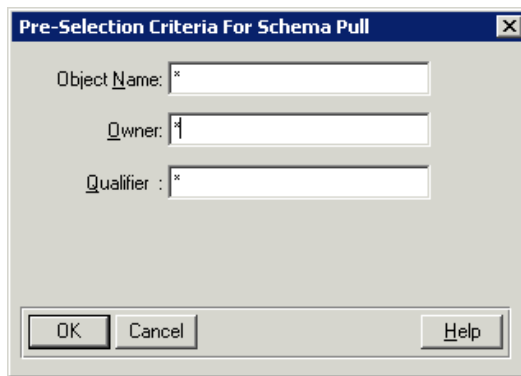
- On the screen that opens, select **Database** □ **Select Working Database...**



- In this dialog, select the name of the SQL Image you are updating (the typical naming convention is the name of the database suffixed by "sql" or "schema").
- Now select **DataServer** □ **dsvtype Utilities** □ **Update/Add Table Definitions...** where dsvtype is the type of dataserver in use. Below is an example using MSS.

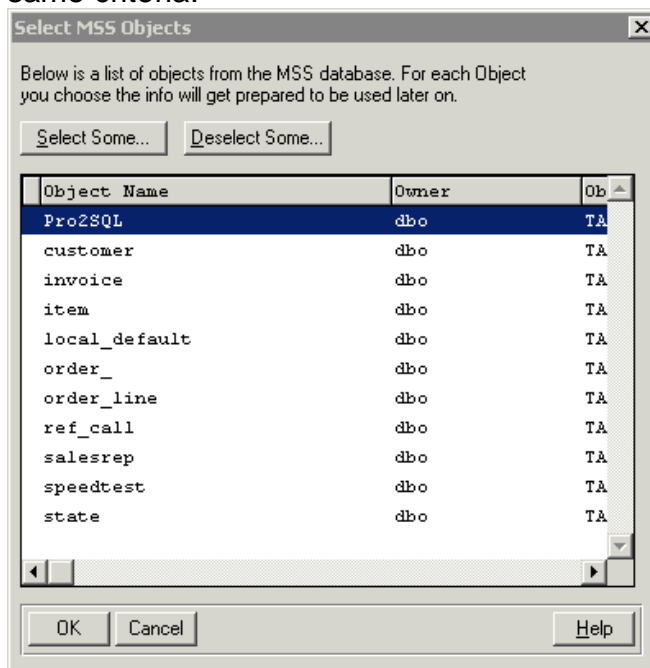


- You will be asked to enter a user-name and password. If you connect to the target database using SQL Server authentication only, you will need to enter the username and password that was created during Pro2 installation. Otherwise, just click **OK**.
- You will then be prompted to pre-select certain schema information.



13. Change Owner to **dbo** and click **OK**.

14. From the next window, you can either select items individually, or you can use the **Select Some...** button to select a group of items fitting the same criteria.



15. Using either method, select those tables that were updated in SQL and click **OK**. Alternately you can do re-pull the ENTIRE contents of the SQL schema by doing;

- Click **Select Some**.
- Enter owner = "DBA" and Object type = "TABLE"
- Click **OK**.
- You should see an asterisk by each TABLE owned by DBO

16. The Data Administration utility will proceed to import the selected tables into the Progress Schema Holder.

17. After the import, you can exit the Data Administration utility and the Progress editor. Your schema holder now contains an accurate image of the current schema from SQL Server.

18. We now need to truncate the before-image file of the schema holder db.

- From the PROENV command line do Proutil dbname -C truncate bi

19. The schema holder has now been updated.

Updating Pro2 Maps

To effectively utilize any changes that have just been made, you must update your Pro2 maps. There are several methods to accomplish this; whichever option you choose, you must account for new tables added (if you want them replicated), new fields added within replicated tables, changed and deleted fields within replicated tables, and replicated tables that no longer exist.

The following steps depict how to update the Pro2 maps:

1. Open the Pro2 Administration Utility, and click on the mapping tab.
2. For each newly added table that you want replicated, select its name in the **Source Tables not mapped** section, and do one of the following:
 - a. If you want to replicate the entire table, click **Auto-Map**, or
 - b. If you do not wish to have the entire table copied, then select its corresponding table name in the **Target Tables not mapped** section, and click **Map Tables**. You will map individual fields as described in the next section.
3. For each replicated table that has new fields that you want replicated, you will need to manually map those fields. Here is the process for mapping those fields:
 - a. Select the particular table you will edit in the **Mapped Tables** browse.
 - b. You will see a list of fields in the lower section of this page.
 - c. For each new field, you need copied, select the field name from the "Source Fields not mapped" section as well as the "**Target Fields not mapped**" section, and click **Map Fields**. These fields will appear in the **Mapped Fields** browse. Alternately you can select the field from the source fields not mapped area and click auto-map.
4. If there are any fields whose data-types, character widths, decimal precision or scales, or names have changed, you will need to delete these fields and re-map them. To delete the field maps, follow these steps:
 - a. In the **Mapped Fields** browse, select the field set you need to update, and click **Delete Field Map**.
 - b. To re-map, follow the steps in step #3.
5. On some occasions, you will have deleted fields from your schema that were previously mapped for replication. These field maps will need to be deleted as well. To do so, select each field set from the **Mapped Fields** table, and click **Delete Field Mapping** as in step #4a.
6. Finally, your schema update may have included dropping some replicated tables. These tables will need to be deleted from the list of mapped tables as well. So, for each table no longer in existence, select the table from the **Mapped Tables** section, and click **Delete Table Mapping**.

Once these steps are completed, your replication setup will be accurate from a mapping perspective. That is, your replication target database will mirror your Progress database. The Progress Schema Image will mirror the replication target database, and the replication maps will be accurate according to the schema

holder and Progress databases.

Regenerating the Replication Code

All that remains in updating Pro2 is actually regenerating code to handle the new structure. This is as simple as selecting two menu options.

Follow these steps:

1. In the Pro2 Administration Utility menu, select **Tools → Generate Code →**
2. **Processor Library**. You will be prompted to select the database.
3. Select the database from the prompt and click **OK**.
4. If you deleted any tables from the map because they no longer exist, or if you added any tables to the map, you will need to regenerate the triggers and mass-copy procedures as well.

In each instance, you will be prompted for the database as above:

- For the triggers, select Tools → Generate Code → Replication Triggers.
- For the Mass-Load procedures, select Tools → Generate Code → Bulk-copy Processor.

The final step in updating the Pro2 system is to perform a Schema update against the Progress database, if you added any tables to replication. This step **MUST** be performed with NO other users in the database. In general, it is strongly recommended that you update the schema in single-user mode. Pro2 includes a stand-alone procedure, `replTriglInsert.p`, which will handle the update in single-user mode; this procedure must be run on the same machine as the database. If you feel that you have ample opportunity to perform the update in multi-user mode, you may attempt to execute this procedure from the Pro2 Administration Utility. Whichever method you choose, once the schema is updated, your changes will be complete. All that is left is to restart replication.

Setting up Pro2 on UNIX

Pro2 provides support for setting up Pro2 on UNIX platforms.

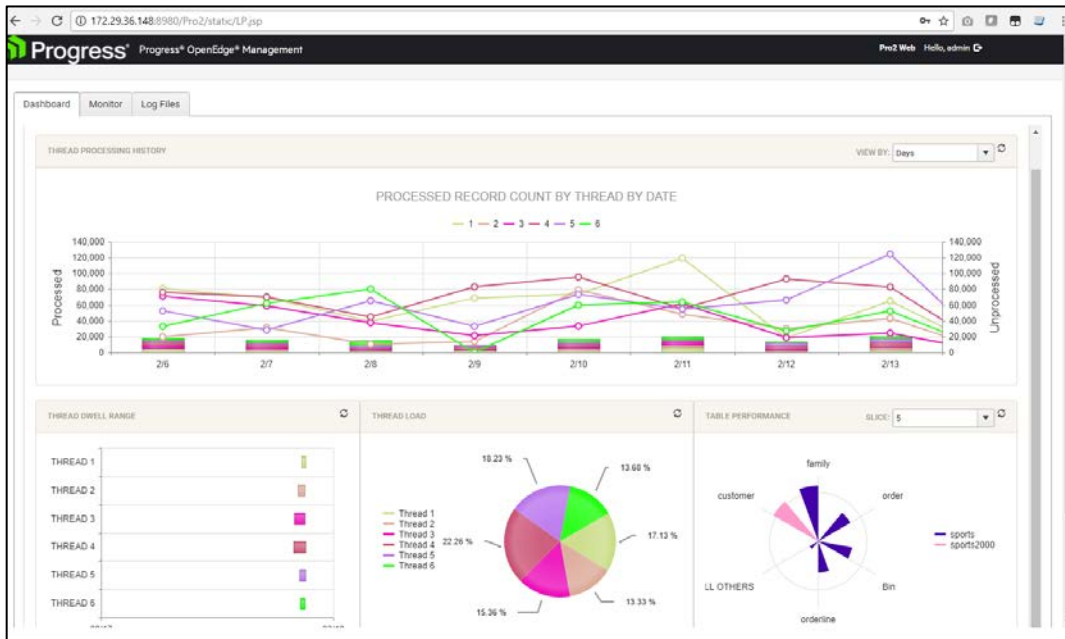
You can run the `pro2pro_menu.sh` script and enter your choice from one of the following options:

1. Load Properties
2. Create DB xref
3. Generate Target Schema
4. Create Mapping
5. Generate Trigger df for db schema
6. Generate Procedure Library
7. Generate Delta DF
8. Quit

When prompted, specify the required input, for example `<dbname>`, to complete the operation.

Configuring and Deploying the Pro2 Web Interface

Apart from the Admin Tool, Pro2 provides a browser-based utility called the Pro2 Web Interface. The Pro2 Web Interface enables users to monitor replication statistics such as processed record counts per thread, table performance, etc.



It provides three tabs:

- **Dashboard**—which displays statistics such as processed record count per thread, thread load, etc., in visually informative charts and graphs.
- **Monitor**—which provides the same functionality as the Monitor tab in the Admin Tool.
- **Log Files**—which displays log entries for each replication thread.

To enable users to access the Web Interface, you need to:

1. Create a replperf database
2. Run the **PASOE_WIN_5_x.bat** file under **Scripts** directory
3. Start the PAS instance and login to Pro2Web

Create a replperf database

This database will contain the replication statistics data that is displayed in the Dashboard tab in the Web Interface.

1. Create the physical database files using the **replperf.st** file located in `<Pro2_HOME>\db\` by running a PRODB command in Proenv. For example:

```
prodb replperf empty
```

Note: Copy the replperf.st file is in the same directory before creating the database. Once that is done, run the command mentioned above.

2. Load the required table definitions using the **replperf.df** file located in <Pro2_HOME>\db\ . You could do this using the OpenEdge Data Administration utility or through the Data Dictionary. To know more about loading data definitions, see:
https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dmadm%2Floading-database-definitions.html%23
3. Finally, start the database using **PROSERVE**.
4. Add the replperf database connection details to the <Pro2_HOME>\bprepl\Scripts\replProc.pf file. So that, replication threads and PASOE instance get connected to replperf database.

Run the PASOE_WIN_5_x.bat file under Scripts directory

Prior to running the PASOE_WIN_5_x.bat file make sure you have updated the Pro2 environment by completing the steps listed below:

- Set the local Pro2 install directory (pro2=c:\Pro2)
- Set the Progress OpenEdge install directory (oe=c:\Progress\OpenEdge)
- Set the name of the PAS instance (rn=Pro2Web)
- Set the location of the .WAR file (wf=c:\pro2\tmp\Pro2.war)
- Set the location if the Progress OpenEdge Work directory (wd=c:\OpenEdge\WRK)
- Set the existing AppServer broker name (bn=restbroker1)
- Set the PAS instance http port (ip=12345)

Once the above steps are taken care of, run the **PASOE_WIN_5_x.bat** file.

Start the PAS instance and login to Pro2Web

Now that all the steps are completed successfully, you are all set to start the PAS instance and login to the Pro2Web interface. To do the same you need to:

1. Start a PROENV session
2. CD C:\pro2\Pro2Web
3. Type bin\tcman start
4. From Pro2 Admin tool, Load the replbasev4.ini file to have the default web credentials
Note: This will override all the existing property values
5. Set USE_WEB_UX to YES in predefs.i file to enable Pro2 web application access
6. Set USE_DATAMART to YES in predefs.i file to capture the replication statistics
7. Login at:

<http://IpAddress:9991/pro2/static/auth/Logon.jsp>

8. Use the default credentials which are admin/admin

Setting colors in the Dashboard

The Pro2 Web Interface uses a default color setting to represent each replication thread in the dashboard. You can specify your own color setting by configuring the WEBBASECOLORS property in the Pro2 Admin Tool. For example, if you have three replication threads, you could specify a comma-separated list of three colors in the WEBBASECOLORS property setting as shown here:

```
#FF0000,#00FF00,#0000FF
```

Creating a copy of Pro2

To create a copy of an existing Pro2 environment in the same machine or a different machine:

1. Create a new **Pro2** folder in your machine.
2. Copy the complete folder structure of the existing Pro2 environment to the new **Pro2** folder.
For example, copy `c:\pro2_demo` to `c:\pro2_prod`.
3. In the new **Pro2** folder, navigate to `bprepl\Scripts` folder and perform the following:
 - a) Edit the **Pro2_env.bat** file and set the value of `PRO2SQL` with the path of the new Pro2 root folder.
 - b) Right-click the **Pro2 - Administration** shortcut file and select **Properties**. In the **Shortcut** tab of the
 - c) **Properties** window, modify the **Start in** field to point to the path of the new Pro2 root folder.
 - d) Repeat Step **b** for **Pro2 - Editor** and **RunBulkLoader** shortcut files.
 - e) Edit all the `.PF` files to point to the corresponding host or port numbers of the databases that are to be
 - f) replicated.

Pro2 Replication FAQ - Frequently Asked Questions

When do I need to regenerate code?

In short, anytime the mapping changes, replication code needs to be regenerated. However, triggers and bulk load processor code only need to be generated 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 is **NOT** updated automatically when schema changes are made to either the source or target database. After changes are made to the target SQL schema and after the schema holder is updated, the

mapping needs to be updated 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. Refer to the **Updating Pro2 after Source Schema Updates** section of this document for more information.

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.

APPENDIX A: Repl Tables Schema Objects

Sequences

Sequence Name	Notes
NextReplNbr	Increments the Sequence Number for each New ReplQueue Record

Tables

The following are a list of Replication DB Tables:

Name	Label
1. ReplControl	Replication Control
2. ReplCustAsgn	Custom Assignments
3. ReplCustDefs	Custom Definitions
4. ReplCustFlds	Custom Fields
5. ReplDBXRef	Database Cross Reference
6. ReplFieldXref	Column Cross-Reference
7. ReplProperties	Properties
8. ReplQueue	Replication Queue
9. ReplTableXRef	Table Cross-Reference

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 defined at this time.

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 CustDefType is Variable, the Data Type. Otherwise the name of the table to which this temp-table or buffer

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 this Field - applies to Decimal and Varchar Fields only
FldDec	Integer	Maximum number of Decimals
FldMand	Logical	Is this field a mandatory field? Yes/No, initial No.

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
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		

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		

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

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
ApplTime	CHARACTER	Time Replication Record was Applied
Audited	LOGICAL	Audited? (Verification)
AudDate	DATE	Date Audited 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

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 #
UseInDiff	LOGICAL	Include in Differentia
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

Appendix B: 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 that 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.
Installation	Initially empty. Used to save installation site-specific files.
misc	Miscellaneous Pro2 utilities
tmp	Miscellaneous Pro2 temporary files used during implementation

Appendix C: Pro2 Program Files

Most of the Progress source files contain a preprocessor definition for `INSTALL_DIR` that needs to be set to the directory path, fully qualified in UNC format that contains the `bprepl` installation directory. This needs to occur before any compile process.

Primary Programs

Program Name	Description
GenReplTrigs.p	Generates the replication triggers. Trigger procedures are created for <code>REPLICATION-WRITE</code> and <code>REPLICATION-DELETE</code> 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 <code>ReplControl</code> sequence in the Replication database. The value is an integer representing the
RunReplProc.p	Initializes the routines for <code>ReplProc.p</code>
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 <code>GenReplTrigs.p</code> , <code>GenReplProc.p</code> , <code>GenSQLSchema.p</code> and
bpadmin.wrx	COM Object support file for <code>bpAdmin.w</code>
ReplAbout.w	Displays version information
About.txt	Includes the versioning text
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
CompReplTrig.p	Compiles the procedure files generated with the <code>GenReplTrigs.p</code> and <code>GenReplProc.p</code> programs.
CompReplProc.p	
UpdSrcSchema.p	Updates the Source database listed in the current database map by creating <code>REPLICATION-WRITE</code> and <code>REPLICATION-DELETE</code> triggers within the meta-schema and then pointing to the
GenSQLSchema.p	Using the current database map, builds a SQL Create Table file in MS-SQL format which will be used to construct a schema in MS-SQL Server that matches the schema of the Progress source database. Generates a .sql file consisting of <code>CREATE TABLE</code> and <code>CREATE [UNIQUE] INDEX</code> 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
GenSQLDiff.p	Using the current database map, builds a SQL file in MS-SQL format that will be 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

LoadFields.p	Used by the Administration Tool to populate the Mapping screen with Fields, corresponding to the selected Table Map, that are not already mapped for replication
LoadTables.p	Used by the Administration Tool to populate the Mapping screen with Tables, corresponding to the selected Database Map, that are not already mapped for replication
MapAllFields.p	Used by the Administration Tool to map all fields associated with a Table that has just been auto-mapped.
MapFields.p	Used by the Administration Tool 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\templ_replproc.p	Template File used by GenReplProc.p to create the processor library.
repl_tmpl\templ_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.
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 bulkload 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

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.

Generated Programs

File Name	Function
<p><ddir>/d<dbName>_<tableName>.p</p> <p>where <ddir> = Delete trigger directory</p> <p>Ex: bprepl\repl_d\dsports_customer.p</p>	<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.</p>
<p><wdir>/w<dbName>_<tableName>.p</p> <p>where <wdir> = Write trigger directory</p> <p>Ex: bprepl\repl_w\wsports_customer.p</p>	<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.</p>
<p><pdir>/<dbName>_replproc.p</p> <p><pdir>/<dbName>_<tableName>_replProc.p</p> <p><pdir>/rp_<dbName>_<tableName>.i</p> <p>where <pdir> = Procedure Library directory</p> <p>Ex: bprepl\repl_proc\rsports_customer.p</p>	<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 will be a "replproc.p" control program for each database.</p>

Appendix D: 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 added to either the session or procedure SUPER stack.

ReplLib.p

- Returns the name of the Kill Event via RETURN-VALUE
- Subscribes to the following events:
 - ReplLibPing**
 - Requires an OUTPUT parameter of type HANDLE
 - Returns the handle of the persistent instance of the library.
 - Used to determine if the procedure is currently running.
 - ReplLibKill**
 - Removes the persistent instance of the library from all SUPER stacks and deletes the library from memory.
 - ReplLogPost**
 - Requires an INPUT parameter of type INTEGER
 - Requires a second INPUT parameter of type CHARACTER
 - Checks the first input parameter against the current Logging Level to determine if a log entry is to be posted
 - If so, posts a log entry into Replication.Log in the Log directory specified by the LOG_DIRECTORY property
- Publishes no events.

Member Functions

FUNCTION 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.

Parameters: INPUT DATE – The date to be formatted

Return: CHARACTER – The formatted date.

FUNCTION formatLogical

Returns the input Logical variable as either a 1 (True) or a 0 (False) in Integer format.

Parameters: INPUT LOGICAL – The logical variable to be converted

Return: INTEGER – a Zero (0) for False values or a One (1) for True values.

FUNCTION getModDate

Takes the input date and performs various manipulations to convert it into a string such as “Weekday, Month dd, yyyy”.

Parameters: INPUT DATE – The date to be converted

Return: CHARACTER – a string interpretation of the date as per Description.

FUNCTION 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).

Parameters: INPUT INTEGER – An integer value representing the elapsed time of the current day, in seconds

Return: CHARACTER – a string interpretation of the time in “HH:MM:SS” format.

PROCEDURE 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 will overwrite current settings.

Parameters: INPUT CHARACTER – a string representing the name of the file from which to load.

PROCEDURE 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 will overwrite current maps with the same cross-reference codes.

Parameters: INPUT CHARACTER – a string representing the name of the file from which to load.



About Progress

Progress (NASDAQ: PRGS) offers the leading platform for developing and deploying mission-critical business applications. Progress empowers enterprises and ISVs to build and deliver cognitive-first applications, that harness big data to derive business insights and competitive advantage. Progress offers leading technologies for easily building powerful user interfaces across any type of device, a reliable, scalable and secure backend platform to deploy modern applications, leading data connectivity to all sources, and award-winning predictive analytics that brings the power of machine learning to any organization. Over 1,700 independent software vendors, 80,000 enterprise customers, and two million developers rely on Progress to power their applications. Learn about Progress at www.progress.com or +1-800-477-6473.

Worldwide Headquarters

Progress, 14 Oak Park, Bedford, MA 01730 USA Tel: +1 781 280-4000 Fax: +1 781 280-4095

On the Web at: www.progress.com

Find us on  facebook.com/progresssw  twitter.com/progresssw  youtube.com/progresssw

For regional international office locations and contact information, please go to www.progress.com/worldwide

Progress, OpenEdge and Corticon are trademarks or registered trademarks of Progress Software Corporation and/or one of its subsidiaries or affiliates in the U.S. and/or other countries. Any other trademarks contained herein are the property of their respective owners.

© 2018 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

ITEM NUMBER