



User Guide

Pro2 Replication Suite

Version 5.5

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.....	3
THE NEED FOR REPLICATION.....	6
REPLICATION OVERVIEW	6
Replication Triggers	7
Replication Processor Flow	7
Pro2 Implementation Options.....	8
Customizable replication	8
Capture Source Changes by SQL Clients	9
Audit Database	9
Logical Deletes	9
WAN.....	10
ARCHITECTURE	10
Pro2 Replication Over a Wide Area Network (WAN)	11
Replication Triggers for LAN (Non-CDC Implementation)	12
Trigger Procedure Generation	12
Processor Library	13
Processor Procedure Generator	13
Processor Library Procedure Generation.....	13
Replication using CDC	14
Splitting CDC Threads	14
Limitations of using Pro2 with CDC.....	15
Replication Processor.....	15
Replication Log Files	15
ReplLog Check Functionality	15
Starting the Replication Processor.....	16
Replication Threads	17
Splitting Replication Threads	17
Notes.....	19
TARGET SCHEMA	19
Target Side Only Tables and Fields.....	19
Support of Customer Target-Only Fields	20
Generating Pro2 specific fields.....	20
Target Side Index Information	20

Standard Modifications to Table and Field Names	21
Standard Data Type Conversions	21
PRO2 ADMINISTRATION TOOL	21
Monitor Tab.....	21
Replication Queue Browser	21
Sleep Interval	22
Refresh Interval	22
Update Buttons.....	23
Replication Status	23
Logging Levels	23
Queue Disposition	24
Replication Status	24
Properties Tab	24
Properties Configuration File.....	24
DB Map Tab	26
Mapping Tab	27
Source Database	27
Source Table/Target Table Viewers	27
Map Tables Button	27
AutoMap Button.....	28
Mapped Tables Browser	28
Options for Mapped Tables	28
CDC Mapping Tab Right-Click Menu	29
CDC Monitor	31
CDC Thread Status	31
Logging Levels	31
Queue Disposition	32
File Menu	32
Tools Menu	32
Generate Code.....	32
Compile Code.....	34
Update Source Schema.....	34
Reset Alarms	34
Reset Bulk Load Control Records.....	34
Run Bulk Loads.....	34
Bulk Load Report.....	35
Custom Retention Rules	36
Generate Encrypted Password	37
Get ReplQueue Count	37
Reassign Queue Thread #	38
CDC Menu	39
CDC Count.....	39

CDC Purge	39
PRO2 WEB INTERFACE	39
COMMAND-LINE REFERENCES.....	41
LOAD_CONFIG	41
SAVE_CONFIG	41
AUTO_DBXREF	41
GEN_TGT.....	42
AUTO_MAP	42
GEN_PROCS	42
GEN_TGT_DIFF	42
TGT_DELTA_DF	43
GEN_DELTA_DF.....	43
CHECK_GEN_Q REC.....	43
UNCHECK_GEN_QREC	44
ADD_THREAD.....	44
DEL_MAP_TABLE	44
DEL_DBMAP.....	45
CHECK_CRC	45
SYNC_REPL.....	45
Run Bulk Loads.....	45
CDC Purge	46

The Need for Replication

The Pro2 Replication Suite was created to provide a lightweight, configurable utility for replicating data from a Progress OpenEdge database to Microsoft SQL Server, Oracle, or to another Progress OpenEdge database. Replication requirements arise for a variety of reasons. Heterogeneous application integration, reporting server, data archival, and business intelligence are just a sample of the business solutions that may require near-real time replication to a foreign database.

Why is Pro2 lightweight? The entire architecture of Pro2 was designed with simplicity in mind. The product was designed using a small footprint with proven technology. The product is written using the **OpenEdge Advanced Business Logic (ABL)**. The process also uses the OpenEdge DataServer as a conduit bridging the gap between OpenEdge fields and data types and the corresponding target fields and data types. The DataServer also includes some useful utilities enabling the two heterogeneous databases to “play nicely” with each other.

How is Pro2 configurable? Built into the design of Pro2 is the ability to replicate one or more OpenEdge databases to one or more target databases. You can duplicate entire databases identically or pick and choose the tables and fields where replication is desired. The source table and field names do not have to coincide with the target table and field names. It should be noted that automatic mapping can take place when source and target table and field names match.

What is ‘near’ real time? Pro2 allows you to configure the time interval between the replication sweeping mechanisms. You can do it every “x” seconds, hourly, twice a day, or daily depending on your specific business problem. It is totally configurable by you and the timing interval can be changed in real time at your will. In addition, the entire replication process can be suspended within the administration tool temporarily. You may also discontinue specific table replication temporarily if needed. This may be done to delay overhead on your production system in a peak period such as end of quarter processing. Once the peak time has passed, simply restart the suspended replication. The only inconvenience in suspending or disrupting the replication process is that users of the target system will not have access to the records that have been updated in your source OpenEdge database until the replication process is restarted and the replication queue is completely swept.

Replication Overview

The data replication process uses a single replication table consisting of minimal data and indexing and a single sequence for process control.¹ No raw data is captured

¹ In the case where Pro2 replication includes auditing, there is a second sequence.

during database events, i.e. record Write and Delete.² As create, update and delete events take place, the replication triggers fire and capture the updated record information to any database replication table (queue). By only capturing database, table, transaction, and event type information, production (replication) overhead is greatly reduced.

All replication data captured is maintained in nine Progress tables. These nine tables can be in a standalone “repl” database or embedded into one of the source Progress databases. A batch process, the “replication processor” will cycle through the replication records periodically, based on user configuration, and replicate the table data directly to the target database via the OpenEdge DataServer.

Replication Triggers

Replication schema triggers are used with the source OpenEdge database. These triggers function as follows:

1. Create the replication record based on the database, table, and ROWID.
2. Record the type of event, i.e. create, write, delete and the transaction number.
3. Set the Applied flag to false (default).

The batch replication processor cycles through the replication records periodically. The time between cycles or “sleep interval” is configurable. The following is the basic functionality of the replication processor.

Note: By default, all replication triggers are compression triggers. However, when auditing is implemented, you cannot use the compression trigger template and you must override the default triggers.

Replication Processor Flow

The Processor Library is a set of procedures used to handle the actual replication of individual records in a given source table over to their corresponding target records.

These procedures are generated during installation or when the user goes to the Pro2 Administration Tool and selects **Generate Code → Processor Library**. These procedures are stored in the **bpreplrepl_proc** folder.

To generate the Processor Library procedures, Pro2 evaluates the field and tables specified for replication and creates logic that will execute writing the data to the target database.

Part of the process of generating the Processor Library is to create an individual “assignment” procedure for each replicated table. This assignment procedure gets stored in the **bpreplrepl_inc** folder using a naming convention of **‘rp_<dbName>_<tableName>.i’**.

² In the case where Pro2 replication includes auditing, raw data is captured when a record is updated.

This solution enables for extensible functionality because users may add customized business logic to these assignment procedures to solve business problems.

For example, if the source records need to be accumulated in some way prior to replication, these assignment procedures can be modified to do so. Obviously, care must be taken if the assignment procedures are edited. If not done correctly, there is risk of breaking replication. Any customized assignment procedures should be backed up in the case that the Pro2 product needs to be reinstalled or reinitialized.

The replication process flow is as follows:

1. Cycle through each replication record where the applied flag is set to false.
 - a. Records will be read chronologically to ensure data updates are consistent from a timing point of view.
 - b. Exclusive locking will be used on each read to insure the update is complete before replication occurs.
2. If the read was successful (Exclusive lock), call the appropriate procedure from the Processor Library for the table number.
 - a. The record is read from the source database.
 - b. The record is read from the target database.
 - i. If the target record is found for the **Write** or **Create** events then the target is deleted and recreated with the source data.
 - ii. If the record is not found for the **Delete** event the Applied flag is set to true.
 - a. Set the **Applied flag** to **True**.
3. If the read failed (Exclusive lock could not be attained): Skip all other replication records with the same transaction number or same ROWID.
4. Pause for the current value of the sleep interval and repeat the replication process.

Pro2 Implementation Options

Pro2 is written completely in the Progress ABL and has been designed for scalability and customization.

Customizable replication

Customizable by Table, by Field/Column, by Record/Row

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

Custom Transformations Using ABL Supported

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

Data Type and Name Transformation

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

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

Table and column names on the target database can be the same or different from those on the source database(s). If they are the same, the Pro2 “Auto-map” function can be used. Otherwise, the tables will need to be manually mapped, however, mapping need only be done once and saved for use in subsequent implementations.

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

Apart from these two, there is another property **GEN_SQL_DEC_FORMAT** which is used to set the format for decimal fields in both schema and differential files. If the property is set to ‘YES’, the format will be same as in the source database. If it is set to ‘NO’ then the default format **[decimal(17,2) null]** will be used. The default value for this property is ‘NO’.

Capture Source Changes by SQL Clients

The standard Pro2 replication triggers capture changes made by ABL (Progress 4GL) clients. Pro2 can also capture changes made to the source Progress database by SQL clients. To do this, java triggers can be implemented in the Progress source database.

Audit Database

In addition to replication, Pro2 can be extended to write to a second target database to maintain audit records. Whereas each row is unique in the primary replication database (via PRROWID column), the audit database will have multiple rows corresponding to each source database record, with a new row created for each change made to the source record.

Auditing is implemented by having a second pass through the repl queue by a separate replication processor designated for the audit database. Typically, when auditing is implemented, a subset of the replicated tables is audited, however, it is possible to audit all tables.

Logical Deletes

There are times when auditing is not required however it is desired to maintain records in the target database after they are deleted from the source database. In this case,

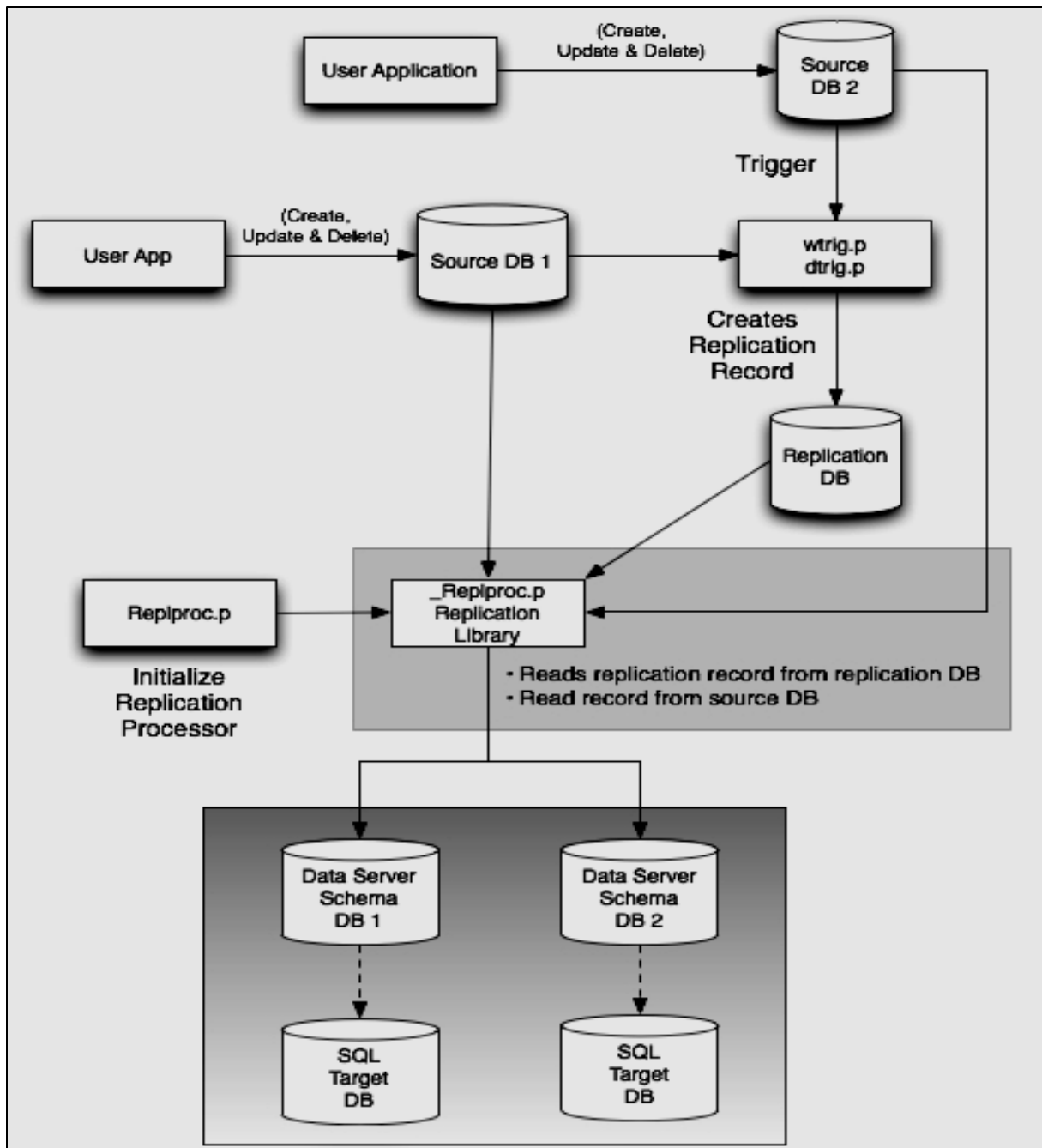
instead of being deleted, rows are designated as “logical” deletes (via PRROWID modification) and the data is kept in the source database.

WAN

Pro2 can be configured for wide-area networks where the source database and target database are in different geographic locations. In this configuration, the Progress AppServer is used on the source database server to communicate across the WAN to the replication processor on the Pro2 server.

Architecture

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



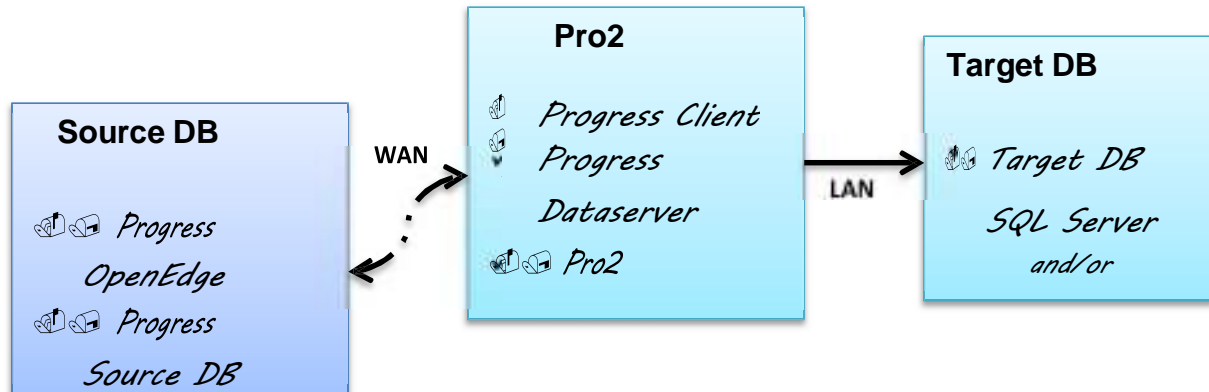
Pro2 Architecture

Pro2 Replication Over a Wide Area Network (WAN)

When the target database is in a different geographic location than the source Progress OpenEdge database, implementation of Pro2 must include an OpenEdge AppServer on the source database server in addition to the standard configuration on the Pro2 server. Instead of running the replication procedures via typical client/server mode connections to the source and repl databases, Pro2 will send

requests to the AppServer on the database server which returns data to the calling procedure on the Pro2 server. The Pro2 server must be on the same local- area- network as the target database server.

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



Note: The Pro2 server can be the same machine as the Target database server. The Pro2 server must run Windows; it may be a virtual machine (VM).

For more information, refer to the **WAN Implementation Guide: Steps and Concepts**.

Replication Triggers for LAN (Non-CDC Implementation)

There are two components to replication triggers. The first is the definition of the replication trigger in the schema of the tables to be replicated that indicates which procedure is to be run whenever there is a Create, Update, or Delete event on a record. In the Progress Software OpenEdge database architecture, the triggers are executed by the client executable (versus the trigger being executed by the database server executable as in most other database systems).

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

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

Trigger Procedure Generation

To generate the replication triggers for tables mapped for replication: from the Administration Tool menu, choose **Tools → Generate Code → Replication Triggers**.

The Trigger Procedure Generator will create REPLICATION-DELETE and REPLICATION-WRITE schema trigger procedures for each replication table. These procedures will be placed in the directories specified by the DELETE_TRIG_DIRECTORY and WRITE_TRIG_DIRECTORY values in the Properties table. The source database and the repl database should be the only databases connected to the session at the time of trigger generation (i.e., the schema holder and target database is not connected at this time; multiple source databases can be connected). The Pro2 Admin Tool will disconnect any schema holders or target databases before generating the triggers.

Processor 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 runs persistently and is added to either the session or procedure SUPER stack.

Processor Procedure Generator

The Processor Procedure Generator creates a single procedure library that contains an internal procedure for each of the replication tables. These internal procedures encapsulate the replication logic specific to a specific table. The resulting procedure library is run persistently while the Replication Processor is running. If this procedure is run from the Progress editor, all databases, i.e. source, schema holder, and target, must be connected. Additionally, the following database aliases will need to be created:

- **SourceDB** for the source database.
- **SchemaDB** for the Progress schema holder database.
- **TargetDB** for the target database (which can be an MSS, Oracle or a Progress database).

Processor Library Procedure Generation

The Replication Processor Library procedures can be generated from either the Administration Tool (**Tools > Generate Code > Processor Library**) menu option or the Progress Editor. The source database can be running in multi-user mode or single user mode.

To generate the library from the Progress Editor:

1. Connect to all four databases, i.e. the Progress source database, the Progress MSS or Oracle DataServer schema holder, the Pro2 repl database, and the target MSS or Oracle database (via an ODBC Data Source). Using the replproc.pf parameter file from the bprepl\Scripts folder will make this easier.
2. Run the SetAliases.p program, using the logical names of your source, schema holder and DataServer target databases as input parameters, to set up the necessary database aliases.
3. Run the GenReplProc.p program.
4. The replproc.p program will be generated and placed in the appropriate directory based on the PROC_DIRECTORY value in the Properties table.

Replication using CDC

Progress OpenEdge provides support for a feature called Change Data Capture(CDC). OpenEdge Change Data Capture is an OpenEdge RDBMS feature that identifies and captures data that has changed in tables of a source database, as a result of create, update, and delete (CUD) operations. Change Data Capture is an industry term that describes the process of duplicating subsets of OLTP data in an external data source with a relatively up to date version of relational data. The OpenEdge implementation of CDC provides a flexible and scalable capture process to facilitate the data extraction, transformation, and eventually the loading of the data to an external data source. Pro2 leverages this feature and replaces the ABL triggers with CDC.

Pro2 CDC implementation also supports a feature called thread#0. When this feature is used, the CDC Admin thread will convert all CDC_change_tracking records irrespective of which thread the table is mapped to. This provides the convenience of having a single CDC thread that converts all CDC records to Replqueue records.

The CDC replication process is faster than using ABL triggers and can be managed online. This results in relatively low database downtime. The client application connection remains unchanged and there is no need to connect to the Repl database. To run multiple threads, users can create a copy of the CDC batch program (CDCBatch.p) and append the file name with the thread number. For example, CDCBatch0, CDCBatch1, CDCBatch2 and so on.

Splitting CDC Threads

You can split each CDC thread into multiple split threads regardless of whether it serves CDC of a single table or group of tables. You can split a single thread into a maximum of 10 split threads.

When you split a thread into n number of parts, it provides n-1 split threads. For example, if you split the thread 5 (CDCBatch5.bat) into four parts, it provides three split threads 51, 52, and 53 along with the original thread 5. This means, the four parts are threads 5, 51, 52, and 53. Pro2 determines the number of a split thread dynamically based on the RECID of the source record, and groups together all the split threads that belong to the same record or RECID. This allows you to run multiple processes without adding queue compression, and prevents out-of-sequence processing of more changes to the same RECID.

Enable/Disable Queue Compression for CDC

You can enable or disable queue compression by means of the CDC_QUEUE_COMPRESSION property. You can add this property by manual intervention from the Pro2 admin tool. In the **Pro2 admin tool**, click **Properties** tab and Select **Add**, Enter the **Property Name** and **Value**. Click **Save**.

If this property exists, and the value is "YES", queue compression is enabled. Else, queue compression will be disabled by default.

Limitations of using Pro2 with CDC

- Pro2 handles only Level 0 policies and policies should be created with the suffix Pro2 to be recognized by Pro2. For example, <tablename>_pro2.
- Pro2 does not replicate any changes that are generated from Non-Pro2 policies.
- Pro2 uses the _userMisc field in the change_tracking_table to determine whether a record is read or not.

Replication Processor

The Replication Processor, also known as the thread, is a Progress batch program that periodically cycles through and processes any Replication Queue records that have not been applied to the SQL database. The name of the thread is the timestamp and the number of the thread. The cycle period is controlled by the sleep interval set in the Monitor Tab of the Pro2 Admin Tool. This value represents the number of seconds between cycles that the processor will “sleep”. If the value is set to 0 (zero) then the Progress batch session will exit and need to be restarted to continue processing of replication records.

Replication Log Files

Each Replication Processor writes to two log files - replproc(YEARMMD-thread#).log and repllog(YEARMMD-thread#).log, that are located in the operating system directory specified by the LOG_DIRECTORY value in the properties table (by default bprepl\repl_log folder). Any errors encountered and general information such as cycle start and stop time and number of records processed will be captured here. These files should be reviewed periodically and will require periodic truncation/deletion or archiving.

When the Logging Level option is set to anything other than “None,” the Replication Processor will generate basic log entries, such as when procedure libraries are executed and when the processor queue is started and stopped. Each level builds upon the previous level. For example, the Moderate setting logs errors as well as the basic entries generated by the Minimum setting. More information about logging levels is found under the **Logging Levels** header in the **Monitor Tab** section.

In addition to the two replication log files, there is an administrative log, Admin.log, that records changes to settings in the Monitor Tab such as sleep interval, logging level, and queue disposition.

ReplLog Check Functionality

The ReplLog Check functionality scans all the log files for any errors that you want to scan and reports the same by means of an email. To execute this functionality, go to \Pro2\bprepl\Scripts folder and run the **RunReplLogChk.bat** file. If you are running this functionality for the first time, it will create the **ErrorTrigger.lst** file (which will be available in the repl_log folder). You can add your error list that needs to be scanned/excluded in this file separated by comma. This action will lead to scanning of all the log files, excluding the unwanted errors and reporting back the error instances found in an email.

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

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

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

Starting the Replication Processor

The Replication Processor is initiated with the following syntax:

```
>_progres.exe -b -pf bprepl\replproc.pf -p bprepl\RunReplProc.p [-param timeout=3600]
```

- **_progress -b** starts a Progress client in batch mode
- **bprepl\replproc.pf** is a parameter file that contains the required database connection parameters
- **bprepl\RunReplProc.p** is the replication processing program
- **-param timeout=3600** is an optional parameter that tells the processor to exit after one hour (3600 seconds)

This command string is also included in the **bprepl\Scripts\replbatch.bat** which can be set to run via Windows Task Scheduler.

The Replication Processor is started from the **bprepl\Scripts\replBatch.bat** file that is typically configured so that it is kicked off by Task Scheduler (Windows) or cron (UNIX). A separate Replication Processor is run for each set of tables grouped by queue thread. The processor is set to run perpetually unless the sleep interval is set to 0 (zero). Setting the value for the sleep interval can be done from the Administration Tool or from the Progress editor (using the **CURRENT-VALUE** statement). The value of the sleep interval represents the sleep or pause time in seconds that the Replication Processor will allow between successive cycles through the unapplied replication queue records. If set to (0) zero, the Replication Processor will require restarting as this will cause the Progress session to exit. If the replication process needs to be “paused” but not “halted” then an adequately large value could be applied to the sleep interval to allow enough time between cycles as needed.

Replication Threads

There can be up to 5 separate threads of replication processors running. For each thread, a separate replBatch.bat from bprepl\Scripts is run (For example: replBatch.bat, replBatch2.bat, replBatch3.bat, etc. corresponding to each thread.) Tables are assigned to specific threads in the **Mapping** Tab of the Pro2 Admin Tool.

Note: To increase the number of Replication threads, please contact your administrator to modify your license and get additional threads.

For most implementations, one thread is typically enough to maintain pace with the changes to be replicated. For implementations with very high transaction numbers, tables can be spread across up to 5 queues to manage performance. Individual tables that have exceedingly high transaction volume or are of very high priority for users of the target database can be assigned their own thread.

Other reasons to have tables in separate threads include running “static” reports on the target database for a subset of tables. These tables are put in their own thread. When it comes time to run the static report, replication for that thread is stopped while the “static” report is run on the target database. Changes for these tables will continue to accumulate. When the report is completed, the replication processor for this thread is restarted and will “catch up” with the held changes in the queue and then continue processing new changes.

Splitting Replication Threads

You can split each replication thread into multiple split threads regardless of whether it serves replication of a single table or group of tables. You can split a single thread into a maximum of 10 split threads.

When you split a thread into n number of parts, it provides n-1 split threads. For example, if you split the thread 5 (replBatch5.bat) into four parts, it provides three split threads 51, 52, and 53 along with the original thread 5. This means, the four parts are threads 5, 51, 52, and 53. Pro2 determines the number of a split thread dynamically based on the RECID of the source record, and groups together all the split threads that belong to the same record or RECID. This allows you to run multiple processes without adding queue compression, and prevents out-of-sequence processing of more changes to the same ROWID or RECID.

Generating new trigger code

To implement splitting of threads, you must generate and install a new replication trigger code. Though you can replace the old trigger scheme for tables or threads that you want to use for splitting, it is recommended and easier to replace all the existing trigger code with the new code generated using a split trigger template.

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

- **repltrig_split_without_compression.p** — Allows splitting of threads but does not apply queue compression.
- **repltrig_split_with_compression.p** — Allows splitting of threads and supports queue compression.

To generate new trigger code:

1. In the Pro2 Admin tool, click the **Properties** tab and modify the property values of the templates **DEL_TRIG_TEMPLATE** and **WRI_TRIG_TEMPLATE** to point to one of the split trigger templates.
2. Click **Tools > Generate Code > Replication Triggers** to regenerate the trigger code.

The new trigger code is generated in the Pro2\bprepl\repl_w (replication writer triggers) and Pro2\bprepl\repl_d (replication delete triggers) folders, and you can use it as required by your application infrastructure.

Enable/Disable Queue Compression

You can enable or disable queue compression by means of the APPSRV_QUEUE_COMPRESSION property. You can add this property by manual intervention from the Pro2 admin tool. In the **Pro2 admin tool**, click **Properties** tab and Select **Add**, Enter the **Property Name** and **Value**. Click **Save**.

If this property exists, and the value is "YES", queue compression is enabled. Else, queue compression will be disabled by default.

Trigger Generation in Lower Case

You can generate the trigger names in lower case by setting TRIGGER_FORCE_LOWERCASE = YES. You can add this property by manual intervention from the Pro2 admin tool. In the **Pro2 admin tool**, click **Properties** tab and Select **Add**, Enter the **Property Name** and **Value**. Click **Save**.

Running split threads

To run the split threads, consider the example of splitting the thread 5 into four parts as mentioned earlier and perform the following:

Note: You must run splitting of threads only after generating the new trigger code.

1. Creating new .bat files:
 - a. Create three copies of the **replBatch5.bat** file located in the **Pro2\bprepl\Scripts** folder and rename them as **replBatch51.bat**, **replBatch52.bat**, and **replBatch53.bat**.
 - b. Edit the **replBatch51.bat** file, locate and modify **Thread-5** to **Thread-51** and **Thread=5** to **Thread=51**, and save your modifications.
 - c. Repeat step 2 for replBatch52.bat and replBatch53.bat files.
2. Creating new thread positions:

- a. To create new thread positions within Pro2, click the Mapping tab in the Pro2 Admin tool and then click **Thread Maintenance**.
The **Thread Change Maintenance** window appears.
- b. For thread 51, type 51 in the field next to the **Create New Thread** button, and then click **Create New Thread**.
- c. Similarly, repeat step 5 for threads 52 and 53.
3. Creating a new property:
 - a. In the Pro2 Admin tool, click the **Properties** tab and then click **Add**.
 - b. In the **Property Name** field, specify the name of the property as **SPLIT_THREAD5**
 - c. In the **Value** field, specify the number of threads as **4**.

Note: The maximum value allowed per thread is 10. However, if you specify a value (characters or negative numbers) other than the numbers 1 to 10, the trigger uses only the actual thread number.

Notes

- Even after generating the trigger code and creating new property for split threads, if you do not create the .bat files and do not schedule the task entries, the data remains in the queue until the new thread jobs are started.
- It is recommended that you do not change the split number if there are existing records in the queue. Changing the split number might push the changes of the same record/rowed to a differently calculated queue.

Target Schema

Pro2 reads the source Progress schema and generates scripts that are then executed on the target database server to create the target schema. By default, all tables and fields are created in the target. The rest of the files (**drop.sql**, **index.sql**, **trunc.sql**, **errors.log** and **warnings.log**) are created separately. These files can be executed by the user as per their requirements. Typically, tables not replicated remain unmapped, however, if it is desired that the tables do not exist at all on the target side, the generated scripts can be edited to remove the pertinent statements.

Target Side Only Tables and Fields

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

- **Pro2sql** – used by Pro2 to verify connection to target database.
- **P2mismatch** – used in customized implementations to monitor column truncation.

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

- **PRROWID** – Unique key that corresponds to the source Progress ROWID.
- **Pro2created** – Date/timestamp record was first created
- **Pro2modified** – Date/timestamp record was last modified
- **Pro2SrcPDB** – name of the source database
- **Progress_RECID** – Used by Pro2 for Progress versions before 10.2

- **Progress_RECID_Ident** – Used by Pro2 for Progress versions before 10.2

Support of Customer Target-Only Fields

Customers are free to add tables, fields and indexes to the target database as needed for reporting or business goals without any impact to Pro2 Replication. However, it is not recommended that unique keys be added as they may conflict with the Pro2 PRROWID unique key. (See further information about indexes below.)

Generating Pro2 specific fields

Pro2 provides an option **Delta DF Pro2Pro** to generate only Pro2 specific fields for your target database.

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

To access this option, click **Tools > Generate Code > Delta DF Pro2Pro** in the Pro2 Administration utility. In the **Generate OpenEdge Target Schema Options** window that appears, perform the following:

- **Generate Pro2 Target Fields** — Select this option to generate Pro2 specific target fields in your target database.
- **Do you want to reactivate the existing inactive index** — Select this option to mark the inactive index as active.

Oracle specialization: You can use the properties listed below for generating Oracle specific output:

- **ORACLE_USE_LOGICAL:** This property is responsible for converting logical field to number(1).
- **ORACLE_USE_SCALE:** This property is responsible for adding precision and scale for decimal fields.

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

Target Side Index Information

Customers can add and delete index information on the target database without interfering with Pro2 Replication as long as no unique keys are added and as long as the PRROWID key is not modified.

Pro2 compiles the CREATE INDEX statements into a separate file named `<dbname>-<targettype>-index.sql` file. These index statements are applied to the target **only if** the customer wants them.

Standard Modifications to Table and Field Names

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

Standard Data Type Conversions

Pro2 performs some data type conversions automatically to get past issues that would typically break other types of ODBC data transfers from OpenEdge to an SQL database server.

- **Date Conversions** – Some versions of MSS and Oracle cannot handle dates with a value before January 1, 1753. To work around this situation, for any dates before 1/1/1753, Pro2 will preserve the month and the date and set the year to 1753.
- **Column Width** – In Progress OpenEdge databases, data is stored in variable length fields regardless of the field width definition. However, in MSS and Oracle, column width is fixed. To work around this, Pro2 will truncate data to the column width defined in the target schema.
- **Logical Fields** – Progress recognizes three possible values for logical fields: True (1), False (0), and Unknown (?). The value “Unknown” is not recognized in foreign databases. To work around this, if a logical value is True, Pro2 will set the target field to True, otherwise it will set it to False.

Pro2 Administration Tool

The Pro2 Administration tool provides a common place to make configuration modifications and monitor the replication process as it runs. The functionality is divided between three tabs; Monitor, Properties, and Tables. The Admin Tool does **NOT** need to be running for replication to occur.

Monitor Tab

Replication Queue Browser

The browser shows the records in the Replication Queue that are pending processing. Records are displayed in chronological order. This is also the order in which the Replication Processor will cycle through the records.

Event Type	Source Database	Source Table	Source Record	Event Date	Event Time	User
C	qaddb	abd_det	0x0000000000000250	10/24/2016	02:11:01	Administrator
C	qaddb	abd_det	0x0000000000000251	10/24/2016	02:11:01	Administrator
C	qaddb	abd_det	0x0000000000000252	10/24/2016	02:11:01	Administrator
W	qaddb	abd_det	0x0000000000000251	10/24/2016	02:11:01	Administrator
W	qaddb	abd_det	0x0000000000000252	10/24/2016	02:11:01	Administrator
W	qaddb	abd_det	0x0000000000000250	10/24/2016	02:11:01	Administrator
C	sports2000	Customer	0x00000000000005fe	10/24/2016	02:11:06	Administrator
C	sports2000	Customer	0x00000000000005ff	10/24/2016	02:11:06	Administrator
W	sports2000	Customer	0x00000000000005fe	10/24/2016	02:11:06	Administrator
W	sports2000	Customer	0x00000000000005ff	10/24/2016	02:11:06	Administrator

Replication Queue Browser

Note that it is possible for the same records to be present in the browser through multiple cycles of the processor. This occurs when the transaction that created the Replication Queue record is still open during the replication cycle and is expected behavior. Replication will only occur on records that have been committed to the Progress database and that are no longer in a locked state. The replication will attempt to retrieve the source table record using the ROWID to verify that the update has completed. If the replication process encounters a record in a locked state, meaning that the update is not yet released or that the source record is being updated again, that replication record will be skipped and processed during the next replication cycle.

Sleep Interval

The screenshot shows the 'Pro2SQL Administration Utility - WAN' window. The 'WAN' tab is selected. Under the 'AppServer' section, the 'Sleep Interval' is set to 5 seconds, with an 'Update' button next to it. To the right, there is a 'Processors Stop: NEVER' label and a 'Change' button. Below this, there are tabs for 'Monitor', 'Properties', 'DB Map', and 'Mapping'. The 'Monitor' tab is active, showing a section titled 'Replication Queue and Options' with a 'Refresh Interval' set to 5 seconds and an 'Update' button.

Replication Processor Sleep Interval

The Sleep Interval value controls the number of seconds the Replication Processor pauses between cycles through the pending ReplQueue records. Changing this value to 0 (zero) will cause the Replication Processor to stop and will require restarting in order to begin processing ReplQueue records again. You must click on the **Update** button to apply changes made to this field.

Refresh Interval

This value sets the automatic refresh rate of the display of Replication Queue records to the value displayed in the Refresh Interval fill-in. The interval is measured in seconds

and will automatically refresh the queue browser at the desired interval. You must click on the **Update** button to apply changes made to this field.

Update Buttons

Sets the corresponding interval (Sleep or Refresh) to the new value entered in the fill-in field.

Replication Status

The Replication Status fill-in shows the status of the Replication Processor as of the last check. The status will be “RUNNING” when the Replication Processor is either actively cycling through the queue or in a paused state between cycles. If the Processor is neither active nor sleeping, then the status will be “STOPPED.” An initial check is performed when the Administration program is first run. Thereafter the check is performed approximately every 3 seconds.

Logging Levels

- A. None – No logging
- B. Minimum – Writes start and stop time of replication process
Writes number of records processed per cycle
- C. Moderate – B plus any errors
- D. Extended – C plus Warnings
- E. Verbose – D plus information for each individual replqueue record processed

Typically, logging level is set to verbose during initial implementation and testing. Logs can grow quite large in verbose mode. To avoid growing the log files too large, you may want to lower the logging level. See the **Pro2 Replication Configuration and Administration Guide** for more information on log file handling.

Global Controls

Replication Control: ☒ On ☐ Off
Logging: MODERATE
Disposition: MARK AS APPLIED

THREAD #	REPL CONTROL	STATUS	LOGGING LEVEL	DISPOSITION
1	ON	STOPPED	VERBOSE	DELETE RECORD
2	ON	STOPPED	MODERATE	DELETE RECORD
3	ON	STOPPED	MODERATE	DELETE RECORD
4	ON	STOPPED	MODERATE	DELETE RECORD
5	ON	STOPPED	MODERATE	DELETE RECORD

Logging Levels and Queue Disposition

Queue Disposition

The Queue Disposition combo-box allows you to select what happens to the replication record after it is processed. Typically, Queue Disposition will be set to Delete Record. When Delete Record is selected, the Replication Processor will delete the replication record once it is processed.

When Mark Record Applied is selected, the Replication Processor will modify the replication record by setting its “Applied” flag to true. No further action is taken on applied records with this setting and a separate PurgeQueue.p procedure needs to be run when the queue records are no longer needed and can be deleted.

Replication Status

When set to on, this option allows the Replication Processor to run. When set to off, this option stops the processor and prevents it from running. To run the processor again, this option must be set back to on, and the processor must be restarted either manually or by a scheduled batch program

Properties Tab

- Add Button -** The Add button enables and clears the Property Name and Value fields for data entry and is used to add a new property.
- Edit Button -** The Edit button is used to modify the value of the currently selected property.
- Delete Button -** The Delete button removes the currently selected property from the Property Browser and from the current configuration.
- Save Button -** The Save button applies all configuration changes to the current configuration.
- Cancel Button -** The Cancel button discards all pending changes and resets the Property Browser to the current configuration state.

Properties Configuration File

The initial settings for all properties are loaded from the “replbasev4.ini” file found in the root Pro2 install folder. This file can be modified manually or via the Properties tab. Below is an example of some of the properties that can be configured.

Property	Description
ADMIN_TITLE	Sets the title of the Pro2 Administration Tool window.
DELETE_TRIG_DIR	Sets the operating system directory that contains the REPLICATE-DELETE schema trigger procedures
LOG_DIRECTORY	Sets the operating system directory where the replication processor log file is located

Property	Description
PROC_DIRECTORY	Sets the operating system directory where the replication processor procedure library is located
WRITE_TRIG_DIR	Sets the operating system directory where the REPLICATION-WRITE trigger procedures are located
REPL_PROC_TEMPLATE	Sets the name of the template procedure used during the creation of the replication procedure library
PROC_TEMPLATE_PRO2SQL	Sets the name of the template procedure that is used during the creation of the replication procedure library for a target SQL database
PROC_TEMPLATE_PRO2ORA	Sets the name of the template procedure that is used during the creation of the replication procedure library for a target Oracle database
PROC_TEMPLATE_PRO2PRO	Sets the name of the template procedure that is used during the creation of the replication procedure library when the source as well as target databases belong to OpenEdge
MASS_LOAD_TEMPLATE_PRO2SQL	Sets the name of the template procedure that is used during the bulk load generation for a target SQL database
MASS_LOAD_TEMPLATE_PRO2ORA	Sets the name of the template generation that is used during the bulk load procedure for a target Oracle database
MASS_LOAD_TEMPLATE_PRO2PRO	Sets the name of the template procedure that is used during the bulk load generation when the source as well as target databases belong to OpenEdge
EXTENT_DELIMITER	GenSQLSchema.p and GenSQLDiff.p use this value as the delimiter for definitions of Progress array fields. Default value: ##

See the description of the Load and Save Configuration menu items under the **File Menu** section below.

DB Map Tab

Before any tables can be mapped from source to target, the source and target database must be specified along with the DataServer schema holder database and the ODBC Data Source.

Pro2SQL Administration Utility - WAN

File Tools WAN CDC Help

AppServer Sleep Interval: 5 Update Processors Stop: NEVER Change

Monitor Properties **DB Map** Mapping

Database Mappings

Tgt Type	Source Database	Target Phys Name	Schema Holder	Schema Image
MSS	mfgdemo	mfgdemo	mfgdemosh	mfgdemosql
MSS	sports	sports	sportssh	sportssql

Pro2 CDC Enabled DB CDC Enabled

Target DB Type: MSS Source Database: mfgdemo

Target Schema Image: mfgdemosql Schema Holder DB: mfgdemosh

Target DB Physical Name: mfgdemo Target ODBC Connection: mfgdemoodbc

Add Edit Delete Save Cancel

DB Mapping Screen

- Source Database:** The logical name of the source database
- Schema Holder DB:** The logical name of the schema holder database
- Target Schema Image:** Name specified in the schema holder for the "Logical Database Name" of the target database
- Target DB Connection:** Name of the ODBC Data Source created for the foreign target database
- Target DB Physical Name:** Name of the target database (for reference only, not used by Pro2)
- Target DB Type:** MSS, Oracle, or Progress

Mapping Tab

Source Database

The Source Database combo-box determines which table cross-references will be displayed in the Mapped Tables browser. It also populates the Source Tables and Target Tables boxes with the appropriate unmapped tables (tables that are not already cross-referenced).

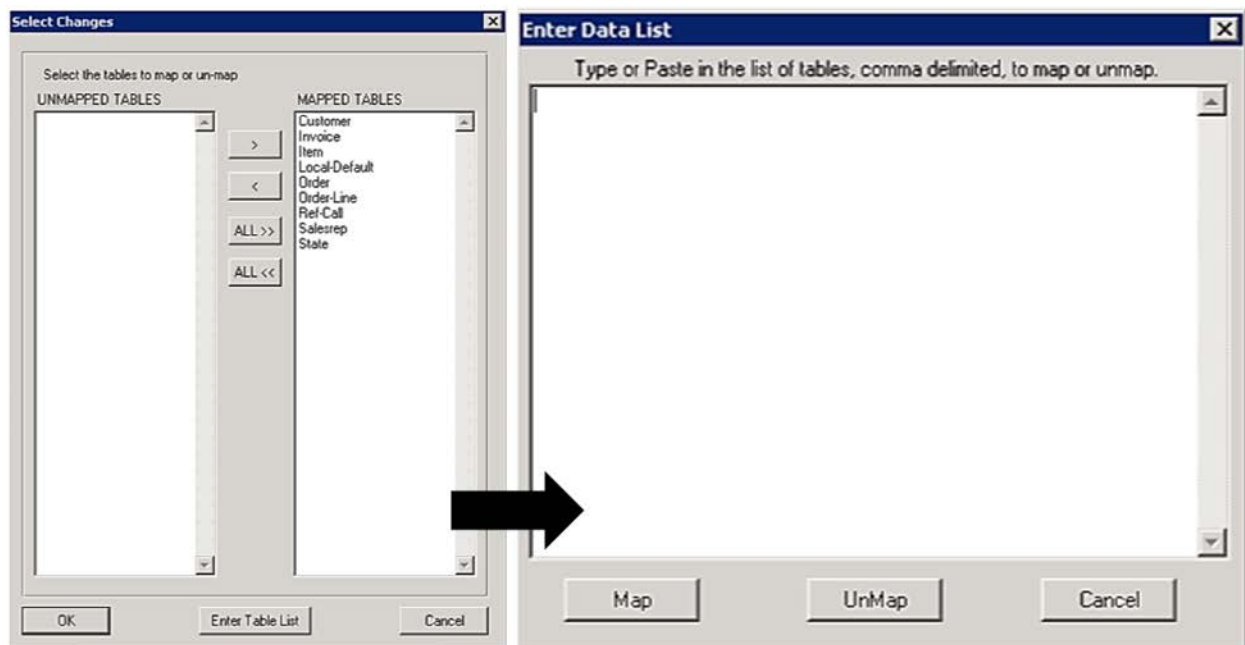
Source Table/Target Table Viewers

These selection boxes contain the table names of those tables in the selected databases (determined by the previous control) that are not already included in the Table Cross-Reference table.

Map Tables Button

The Map Tables button takes the selected table from the Source Tables box and the selected table from the Target Tables box, and creates a Table Cross-Reference record for them.

It also enables the users to map and unmap specific tables by taking table names as the input from the database. Clicking on **Map Tables** opens the **Select Changes** window, where you click **Enter Table List**. This opens the **Enter Data List** window where you can type or paste a comma separated list of table names and choose to **Map** or **Unmap** these tables from the Data List.



Map Tables

AutoMap Button

As an option, you can select multiple tables within the Source Tables box and press this button, and the application will automatically map those tables with matching target tables. In addition, for those tables that do get auto-mapped, the application will continue and create Field Cross-Reference records for any matching fields of those tables. Note that this only works for those tables and fields that have a direct matching table name or field name in the corresponding target.

Mapped Tables Browser

This browser displays all tables currently cross-referenced under the database cross-reference selected by the Working Database Map. Scrolling through this browser changes the display of associated fields.

Options for Mapped Tables

<input type="checkbox"/> No Policy	Gen Queue Record	Process Queue Rec	Include/Exclude Diff
<input type="checkbox"/> Inactive Policy	<input checked="" type="checkbox"/> Generate Queue Rec	<input checked="" type="checkbox"/> Process Queue Rec	<input checked="" type="checkbox"/> Include in Differential
<input checked="" type="checkbox"/> Active Policy	Map Tables	Thread Maintenance	Add/Remove Policies
			(In)Activate Policies

Replication Administration Screen – Mapped Tables Options

Generate Queue Record

Uncheck this box to stop generating replqueue records for the selected table (i.e., turn off replication but leave triggers in schema). It will set ReplTableXref.genqrec to FALSE for this table. Default is checked. If it is desired do this for an entire database, the following ABL code can be run from the Progress Editor to stop generating queue records for all tables: FOR EACH ReplTableXref: genqrec = FALSE. END.

If Generate Queue Record is off, replication queue records will be skipped for that table even if Process Queue Records is on.

Process Queue Record

Uncheck this box so that any Replqueue records generated will not be processed (i.e., replicated). Queue records will stay in queue until this is checked back on. Default is checked. Works same as the option for Generate Queue Record but sets ReplTableXref.procqrec to FALSE.

Include in Differential

Default is checked. If this is unchecked, any changes to the schema for this table will not be included in the Differential Schema generated by **Tools->Generate Code->Target Differential**.

Queue Thread

This specifies which thread the table will be in. Mapped tables can be spread amongst up to five separate threads. By default, all mapped tables are put in queue 1. See the **Replication Queue Threads** under **Replication Processor** in the **Architecture** section for more information.

Merged Triggers

This will be checked automatically if there was a pre-existing replication-delete and/or replication-write trigger on the table (RepTableXref.mrgdtrig). If so, the Pro2 trigger needs to be merged with the pre-existing trigger.

Source Fields/Target Fields Viewers

These selection boxes act just like the corresponding table viewers. They contain the names of fields, based on the selected table cross-reference as determined by the Mapped Tables browser, that are not currently mapped to one-another.

Map Fields Button

The Map Fields button takes the selected field from the Source Fields box and the selected field from the Target Fields box, and creates a Field Cross-Reference record for them. In addition, it obtains additional information, such as data-types and width/decimal information.

Mapped Fields Browser

This browser displays all fields currently cross-referenced under the selected record in the Mapped Tables browser. Scrolling through the browser displays additional information in the fields to the right of the Fields section.

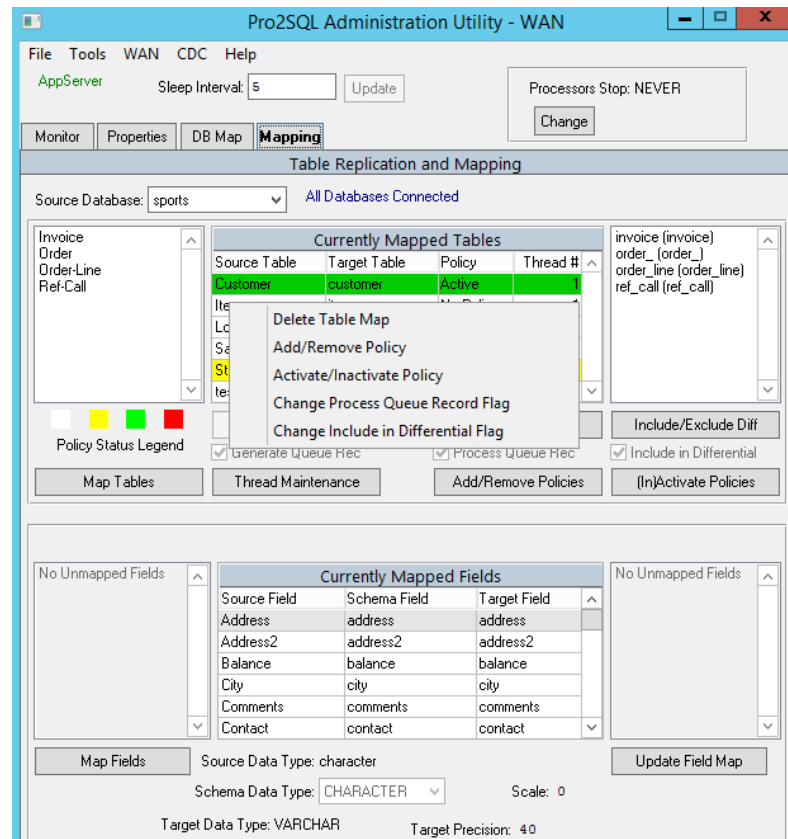
Deleting Table Mapping and Field Mapping

You can delete mappings for tables and fields by right clicking on the corresponding tables or fields and click **Delete**.

Alternatively, you can click **Map Tables** OR **Map Fields** button to unmap the respective tables and/or fields.

CDC Mapping Tab Right-Click Menu

Users can perform the following options on the Mapping tab of a CDC enabled setup. Right click on one of the items in the Currently Mapped Tables and select a menu item to perform the corresponding action.



Replication Administration Screen – Mapped Tables Options

The menu items are explained as follows:

Delete Table Map

This menu item can be used to delete a Table map.

Add/Remove Policy

This menu item can be used to either add or remove a CDC policy to an existing table map. After adding a policy, use the [IN]Activate Policies button to activate a policy.

Activate/Inactivate Policy

This menu item can be used to activate or de-activate a policy for previously mapped table.

Change Process Queue Record Flag

This drop-down menu item can be used to alternate between the Yes and No values for whether a process queue record should be flagged

Change Include in Differential Flag

This drop-down menu item can be used to change whether a differential flag is set to include or exclude a change record.

CDC Monitor

The CDC Monitor menu item can be used to show the status of the get CDC change tracking table as of the last check. The status will be "RUNNING" when the CDC Processor is either actively cycling through the queue or in a paused state between cycles. If the Processor is neither active nor sleeping, then the status will be "STOPPED."

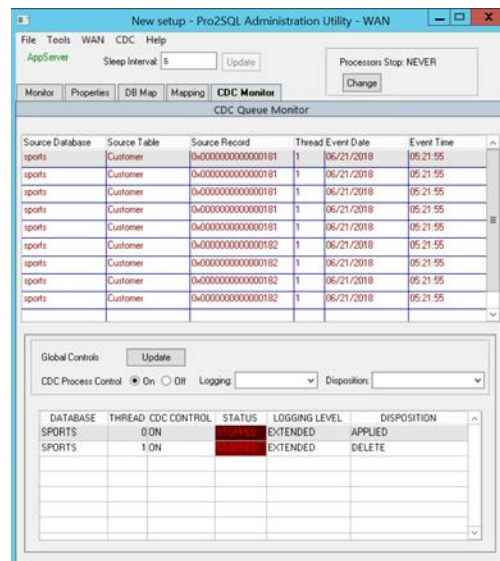
CDC Thread Status

You can check the status of the CDC threads running on the AppServer side by means of the **APPSRV_CDCPROC_SRCSIDE** property. You can add this property by manual intervention from the Pro2 admin tool. In the **Pro2 admin tool**, click **Properties** tab and Select **Add**, Enter the **Property Name** and **Value**. Click **Save**.

If this property exists, and the value is "YES", the thread status will be visible. Else, it will be disabled by default.

Logging Levels

- A. None – No logging
- B. Minimum – Writes start and stop time of replication process
Writes number of records processed per cycle
- C. Moderate – B plus any errors
- D. Extended – C plus Warnings
- E. Verbose – D plus information for each individual replqueue record processed



Logging Levels

Typically, logging level is set to verbose during initial implementation and testing. Logs can grow quite large in verbose mode. To avoid growing the log files too large, you may want to lower the logging level.

Queue Disposition

The Queue Disposition combo-box allows you to select what happens to the CDC record after it is processed. Typically, Queue Disposition will be set to Delete Record. When Delete Record is selected, the CDC Processor will delete the CDC record once it is processed.

When Mark Record Applied is selected, the CDC Processor will modify the record by setting its “Applied” flag to true. No further action is taken on applied records with this setting and a separate PurgeQueue.p procedure needs to be run when the queue records are no longer needed and can be deleted.

File Menu

Load Configuration	Loads list of replication control properties and values from a text file.
Save Configuration	Saves the current list of replication control properties and values to a text file.
Load Map File	Loads table and field mapping cross-reference information from a text file.
Save Map File	Saves current table and field mapping cross-reference information to a text file.

See the **Properties Tab** section for more information on the configuration file and properties settings.

Tools Menu

The Tools menu in the Pro2 Administration utility provides the following options:

Generate Code

The Generate Code Sub-Menu consists of the following choices:

Replication Triggers

The Replication Triggers menu item will run the procedure that creates the replication schema triggers. All conditions mentioned in the section describing the Replication Trigger Generator section of this document must be met before this procedure can be executed.

Processor Library

The Processor Library menu item will run the procedure that creates the Replication Processor Library procedures.

Bulk-Copy Processor

The Bulk Copy menu item generates the bulk-copy processor procedures. These procedures will perform a mass-record copy of the entire source database over to the DataServer target.

Target Schema

The Target Schema menu item generates the necessary database script to build a copy of the Progress schema within the target database server. You are prompted for an output file name and then asked for the source database from which you wish to prepare a database Schema. The resulting DB file can then be executed within an empty database container to build a target database with the same schema as the source Progress OpenEdge database.

Target Differential

The Target Differential menu item generates the necessary queries to upgrade the current target schema to match to current source schema. Typically, this utility is used after a schema change to the source database. The resulting database file can then be used to upgrade the target schema to match the source schema.

Note: Anytime a change is made to the schema of the target database you must rebuild the schema holder database before Pro2 will see the changes to the target schema.

Oracle specialization: You can use the properties listed below for generating Oracle specific output:

- ORACLE_USE_LOGICAL: This property is responsible for converting logical field to number(1).
- ORACLE_USE_SCALE: This property is responsible for adding precision and scale for decimal fields.

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

For Oracle specific targets, you can make use of the properties listed below. These properties are responsible for Add support for precision and scale.

Trigger Delta DF

In WAN environment, Pro2 connects to a local empty copy of the source DB and generates a Delta DF from the DF of the source DB by removing all area information from the source DF and downloading it through the AppServer.

Linked Server SQL

You use this menu item to generate SQL scripts containing INSERT INTO statements that you can execute on a Microsoft SQL Server database. This is a faster alternative to running bulk loads. If you choose this method, you must

provide the name of a Linked Server in Microsoft SQL Server and also the name of an OpenEdge ODBC data source.

Compile Code

The Compile Code sub-menu consists of three options: Replication Triggers, Processor Library and Bulk-Copy Processor. These options will compile the procedures created by the “Generate Code” sub-menu counterparts and save the compiled .r code in the same directory as the source .p code.

Update Source Schema

The Update Source Schema menu item will install the generated (or compiled) Replication Triggers into the source database indicated. **NOTE:** As this command actually impacts the meta-schema of the source database, the database must be running in either single-user mode (no other users connected) or with no open active transactions running against it.

Reset Alarms

This menu item will reset the alert flag. The alert flag is set if replication status checks are configured and an alert was triggered in the event the status check found replication not running.

Reset Bulk Load Control Records

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 select the databases that you want to include for this action and click OK.

Run Bulk Loads

This menu item will start the Bulk Load Utility that will perform a mass-record copy of the entire source database over to the DataServer target. You can also choose to bulk load only a few select tables based on the number of records they contain. For example, to begin with, you may want to bulk load only those tables that have over 100,000 records. Or, alternatively, you may want to bulk load only those tables that have less than 100,000 records. You can accomplish this by performing a database or table analysis using the DBANALYS or TABANALYS command in proutil and then using the output to create filters in the Bulk Load Utility.

The **Save Settings** button saves all the information provided in the **Bulk Load of Tables** window. This saves the effort of feeding in the same information again.

This feature is described in greater detail in the *Pro2 Configuration and Administration Guide*. To know more about running a database analysis, see:
https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dmadm%2Fdatabase-analysis-output.html

Run Bulk Loads

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 is 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: This operation can be applied for one table only. It can't be applied on multiple tables. You also need to ensure to run this operation once you have upgraded replmbatch.bat/sh file in bprepl/Scripts folder.

Bulk Load Report

This menu item will generate a Bulk Load report after the Run Bulk Loads command is executed. You can generate the report by clicking **Tools** and selecting **Bulk Load Report** on the top ribbon of the **Pro2SQL Administration Utility window**. This command will take you to the **Bulk Copy Results** window where you can select the table for which the Bulk report needs to be generated and click **Run**. The Bulk Copy Report will be generated. You can also choose to save the report in \bprepl\repl_log

folder by clicking the **Save to File** option. The report will be saved in form of an excel sheet in the target folder.

Here is an example of the Bulk Load Report:

Bulk Copy Report

Save to File

Loaded	Incomplete	Not Loaded	File Name	Action	DB Table	Status	Date time	Rows Loaded	Rows Locked
1	0	0	Customer	BULKCOPY	sports.Customer	COMPLETE	05/08/2018-	83	0
2	0	0	Invoice	BULKCOPY	sports.Invoice	COMPLETE	05/08/2018-	147	0
3	0	0	Local-Default	BULKCOPY	sports.Local-Defaul	COMPLETE	05/08/2018-	10	0
4	0	0	Order	BULKCOPY	sports.Order	COMPLETE	05/08/2018-	207	0
5	0	0	Ref-Call	BULKCOPY	sports.Ref-Call	COMPLETE	05/08/2018-	13	0

Custom Retention Rules

This menu item allows you to maintain the custom data retention rules used on specific tables. These rules are applicable in Pro2SQL for tasks like Bulk-Load, Bulk-ASCII-Export, Verification, and Replication. An IF statement is used to exclude Data that satisfies the condition. To set custom retention rules, take the following steps:

1. From the Tools menu, select the **Custom Retention** menu item. The Custom Retention window opens.
2. To specify a new Custom Retention rule for a specific table, click **Add**.
3. For example, if the DB is sports2000 and the rule you want to set is to not replicate any customers who have country set to USA, enter the details as shown in the following image:

Custom Data Retention Rules - Domain:New setup

Custom Rule

[Help]

Source DB: Sports

Source Tbl: Customer

Type: Retention_Rule

Condition Statment: <<Source_DB>>.Customer.Country = "USA"

Past-DB-TAB

First Prev Next Last Update Add Delete Save Cancel Quit

Enter data or press ESC to end.

Custom Data Retention Rules

4. Click **Save**.

Notes:

- For rule changes to take effect in the WAN setup, you should run **WAN->Build AppSrv Processor Libraries** from Admin tool.
- For rule changes to take effect in the LAN setup, you should generate the **Process-Library** and **Bulk-Copy-Processor**.
- DB fields should be reference in full – '~{DB~}=~{Table~}.~{field~}' where '~{DB~}'=<<SOURCE-DB>>.
- Always use <<SOURCE-DB>> irrespective of whether you are referring to Source or Target. The system will select the right DB.

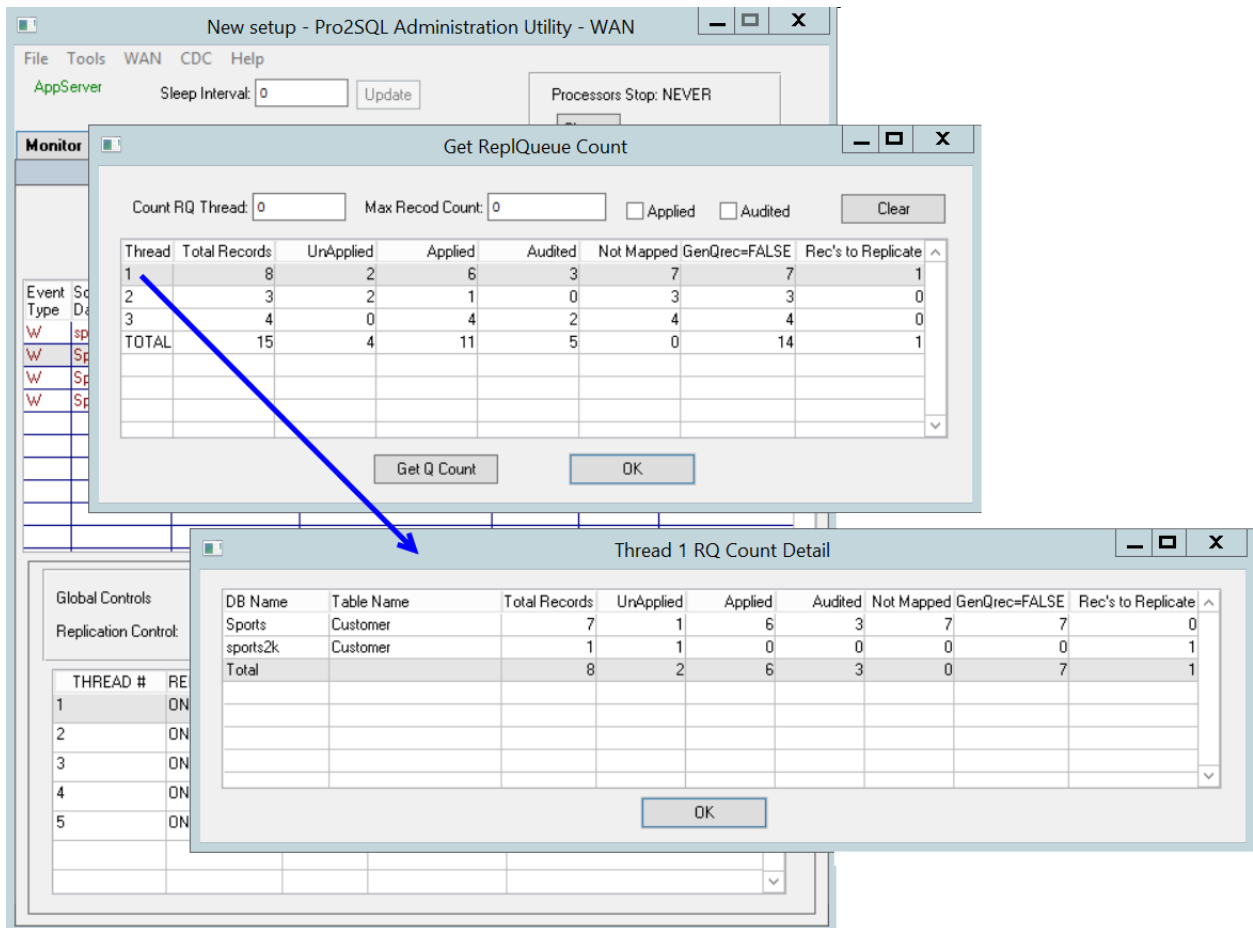
Generate Encrypted Password

This menu item is used while setting up a user for a secure database connection. To set up the user name, password and the domain, select this menu item and enter the details and create OK.

Get ReplQueue Count

The Get ReplQueue Count menu item enables you to get the thread count for a specific Repl Queue thread. Along with the thread number, you can specify the maximum number of records of the thread you want to view, and choose whether you want to view applied or audited records, or both. To get the count for all the threads, enter '0'.

Upon choosing your options, click the **Get Q Count** button to view the total number of records, and the number of unapplied, applied, audited, and unmapped records available for that thread. You can right-click on the thread and click **Output to File** to save these details as a **RQ_count.txt** file located in the bprepl\repl.log folder. In addition, you can double-click a thread to view a list of databases, tables, and records associated to it, as shown below.

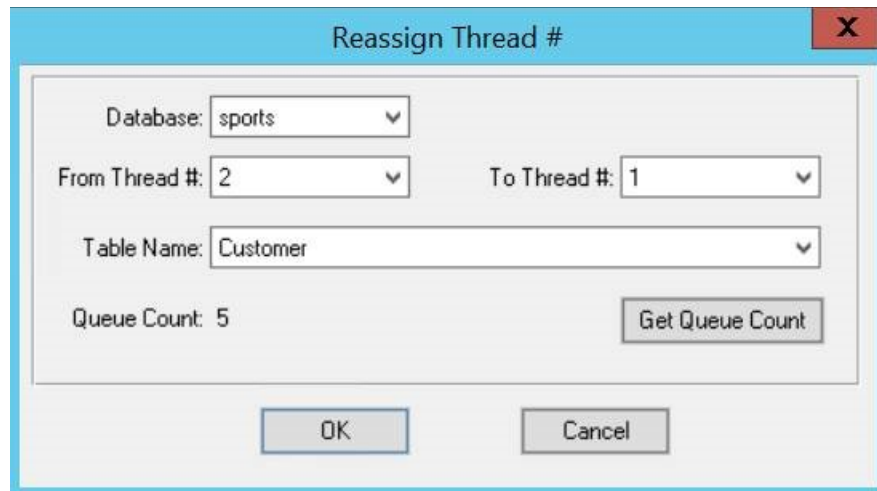


Get ReplQueue Count

Reassign Queue Thread

The **Reassign Queue Thread #** menu item enables you to reassign the queue records from one thread to another. To reassign the queue threads, Click **Tools → Reassign Queue Thread #** in the **Pro2 Admin** window. The **Reassign Thread #** window will open. Here you can choose thread number to assign the queue records to. You can also get queue count by clicking on the **Get Queue Count** button. This functionality enables you to get the queue count based on the input provided. The Reassign Queue Thread # functionality also allows you to choose the database and the table name as well.

Note: The Reassign Queue Thread # will not be complete until all the input items are selected.

A screenshot of a 'Reassign Thread #' dialog box. The dialog has a title bar with a close button (X). Inside, there are several input fields: 'Database:' with a dropdown menu showing 'sports'; 'From Thread #:' with a dropdown menu showing '2'; 'To Thread #:' with a dropdown menu showing '1'; 'Table Name:' with a dropdown menu showing 'Customer'; and 'Queue Count:' with the value '5'. There is a 'Get Queue Count' button next to the Queue Count field. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

Reassign Queue Thread #

CDC Menu

To enable the CDC feature and the corresponding menus, users should set the Enable_OE_CDC preprocessor to YES in the predefs.i file. These tasks can also be performed using the command line utilities for both WAN and LAN. Refer to the Command Line Support for Admin Tool Operations section for more information.

CDC Count

The CDC Count menu item can be used to get the actual record count of the CDC change tracking table based on whether the records are processed or not processed. This can be selected using the dropdown menu. This can be done by selecting the Applied/Not Applied dropdown from **CDC>CDC Count**. Users can get the processed records by selecting **Applied** from the **Operation Type** drop down. To get the list of records that are not processed, users should select **Not Applied** from the **Operation Type** drop down.

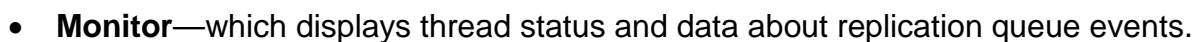
CDC Purge

The CDC Purge menu item can be used to delete the data of change records other than the old data based on user input about the number of days of data that should be deleted and the maximum number of records that should be deleted per batch. Max processed per batch. To delete change data records, from **CDC>CDC Purge**, select the source database and the number of days of data that you want to retain. Specify the number of records to be processed per batch and click **Submit**. If the CDC_PURGE_DAYS property is not specified, this menu item will be disabled.

Pro2 Web Interface

Apart from the Administration Tool, Pro2 provides a browser-based utility called the Pro2 Web Interface, which you can use to monitor your replication statistics.

- **Dashboard**—which displays statistics such as processed record counts for replication threads and table performance in graphs and charts.



Pro2 User Guide

WEBBASECOLORS property setting as shown here:

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

Note that before you can access the Pro2 Web Interface, the Web Interface's web app needs to be configured and deployed to a Progress Application Server for OpenEdge instance. The steps to configure and deploy the web app are detailed in the *Pro2 Configuration and Administration Guide*.

Command-line references

Pro2 supports command-line operations for some of the Admin tool operations. This is an alternative to performing the tasks using command prompt instead of the Pro2 Admin tool user interface.

This section details the different modes and their respective syntax to execute the admin tool operations. These operations can be execute using the BatchGen.bat file located in the Scripts folder of your Pro2 installation.

LOAD_CONFIG

Loads the configuration file from the location given using PATH variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\replbasev4.ini;DB=All;Mode=LOAD_CONFIG " >>
%CODEDIR%\repl_log\batch.log
```

Parameters

SAVE_CONFIG

Saves the current configuration to the path given in PATH variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= SAVE_CONFIG" >> %CODEDIR%\repl_log\batch.log
```

Parameters

AUTO_DBXREF

Adds the DB map record to the Database Mappings table.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"Mode=AUTO_DBXREF;DB=sports,Options=MSS "> %CODEDIR%\repl_log\batch.log
```

Parameters

GEN_TGT

Generates Target schema to the location given in PATH variable

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_TGT Options=NO|YES|NO " >>
%CODEDIR%\repl_log\batch.log
```

Parameters

AUTO_MAP

Maps the tables listed in the Mapping table automatically

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= AUTO_MAPP " >> %CODEDIR%\repl_log\batch.log
```

Parameters

GEN_PROCS

Generates the Process libraries, Replication triggers, Bulk-copy processors to the default location specified in properties.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_PROCS " >> %CODEDIR%\repl_log\batch.log
```

Parameters

GEN_TGT_DIFF

Generates the target differential files and saves them to the location given in PATH variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_TGT_DIFF;Options=NO|YES|NO " >>
%CODEDIR%\repl_log\batch.log
```

Parameters

TGT_DELTA_DF

Generates the trigger Delta DF file to the location given in the PATH variable

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= TGT_DELTA_DF " >> %CODEDIR%\repl_log\batch.log
```

Parameters

GEN_DELTA_DF

Generates a Delta DF file on the WAN side and saves the files to the location given in the PATH variable.

Syntax

```
"%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode= GEN_DELTA_DF " >> %CODEDIR%\repl_log\batch.log
```

Parameters

CHECK_GEN_QREC

Enables the functionality of generating replication queue records for single and multiple tables (which are separated by comma (,)).

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode=CHECK_GEN_QREC;TABLE=Customer" >> %CODEDIR%\repl_log\
batch.log
```

Parameters

TABLE

Name of the mapped table. You can specify a single table name, multiple table names

or ALL.

UNCHECK_GEN_QREC

Disables the functionality of generating replication queue records for single and multiple tables (which are separated by comma (,)).

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode=UNCHECK_GEN_QREC;TABLE=Customer" >>
%CODEDIR%\repl_log\batch.log
```

Parameters

TABLE

Name of the mapped table. You can specify a single table name, multiple table names or ALL.

ADD_THREAD

Adds a new replication thread.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode=ADD_THREAD;THREAD_NUM=10"
>> %CODEDIR%\repl_log\batch.log
```

Parameters

THREAD_NUM

A number for the thread you are adding. The value must be an INTEGER ranging between 1 and 99.

DEL_MAP_TABLE

Deletes a mapped table along with its mapped fields for single and multiple tables (which are separated by comma (,)).

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param
"PATH=C:\Pro2\;DB=All;Mode=DEL_MAP_TABLE;TABLE=Customer" >> %CODEDIR%\repl_log\batch
.log
```

Parameters

TABLE

Name of the mapped table. You can specify a single table name, multiple table names or ALL.

DEL_DBMAP

Deletes the mapped records of a database.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param "PATH=C:\Pro2\;DB=sports;Mode=DEL_DBMAP" >>
%CODEDIR%\repl_log\batch.log
```

Parameters

DB

Name of the database. You can specify a single database name or ALL.

CHECK_CRC

Checks whether the CRC matches for mapped tables in the source databases and the local databases in a WAN environment.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param "PATH=C:\Pro2\;DB=All;Mode=CHECK_CRC" >>
%CODEDIR%\repl_log\batch.log
```

Parameters

DB

Name of the database. You can specify a single database name or ALL.

SYNC_REPL

Synchronizes the replication database table values in a WAN environment.

Syntax

```
"%PROEXE%" -ininame Pro2.ini -basekey "INI" -b -pf %REPLPF% -p
%CODEDIR%\BatchGenerators.p -param "PATH=C:\Pro2\;DB=All;Mode=SYNC_REPL" >>
%CODEDIR%\repl_log\batch.log
```

Parameters

None

Run Bulk Loads

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

Syntax

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

Parameters

Parameter	Description
TableInc	Comma delimited list of table names to be loaded. If not included on the param line, the program defaults to "*" (all tables)
TableExc	Comma delimited list of table names to be EXCLUDED from the load. If not included on the param line, the program defaults to "xxxxxxx" (no tables to be excluded)
ResetBulkload	Used to determine if the load should reload any tables previously marked as complete. Valid entries are "Yes" or "No"
ResetInProcess	Used to determine if the load should restart from scratch any tables with a status of "in process". 'Yes' means restart from scratch, No means attempt to restart where previously stopped
DB	Point this to a mapped source db. Leave blank or remove for the -params line to load all mapped databases. Invalid entries will cause the load to abort.
RunFlag	Determines the "run flag" to be used. Defaults to RunFlag="A" then not send in via command line parameters.
ForThread	If entered the process will only load tables mapped to the ForThread value. Default is ForThread=0. Load tables for any thread number.
MProcThreads	Used to determine how many "threads" this runFlag process will use for this load. Default is MProcThreads=9. Maximum Value = 9, Values greater than 9 will be reset to 9.
MaxRecordCount	Primarily used for testing. If a value is entered, the load for each table will stop once the MaxRecordCount is reached. Default is MProcThreads=0. (Do not stop for max record count).

CDC Purge

The CDC Purge menu item can be used to delete the data of change records other than the old data based on user input about the number of days of data that should be delated and the maximum number of records that should be deleted per batch.

Syntax

```
%PROEXE%" -b -pf %REPLPF% -p %CODEDIR%\CDCPurge.p -param
"DB=sports,CDC_PURGE_DAYS=0,MAXPROCESSED=100" >
%CODEDIR%\repl_log\CDCPurge.log
```



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