|  |
| --- |
|  |

|  |
| --- |
| **Replication, Backup and Remote Update** |
| www.sieradelta.com |
|  |
| User manual for Replication, Backup and Remote Update. |
|  |
| **Simon Carter** |
| **2/13/2017** |
| **Copyright © 2017. SieraDelta.com All Rights Reserved.** |

# Revision History

|  |  |  |
| --- | --- | --- |
| Version | Date | Status and Description |
| 0.01 | 13/02/2017 | Initially created. |

[Revision History 1](#_Toc475004621)

[Replication 3](#_Toc475004622)

[Advantages of replication 3](#_Toc475004623)

[Disadvantages of replication 3](#_Toc475004624)

[Before You Begin 4](#_Toc475004625)

[Steps Required for Replication 4](#_Toc475004626)

[Steps Required to Remove Replication 4](#_Toc475004627)

[Change Logs 4](#_Toc475004628)

[Considerations for Replication 5](#_Toc475004629)

[Site ID 5](#_Toc475004630)

[Generators 5](#_Toc475004631)

[Unique Master ID 5](#_Toc475004632)

[Unique Site ID 5](#_Toc475004633)

[Configuration 6](#_Toc475004634)

[Configure Database 7](#_Toc475004635)

[Options 7](#_Toc475004636)

[Current Database 8](#_Toc475004637)

[Master Database 9](#_Toc475004638)

[Replication Settings 10](#_Toc475004639)

[Tables to Replicate 11](#_Toc475004640)

[Colour chart 11](#_Toc475004641)

[Sort Order 12](#_Toc475004642)

[Menu Options 12](#_Toc475004643)

[Configure Table 13](#_Toc475004644)

[Replication Add Schema 14](#_Toc475004645)

[Replication Remove Schema 15](#_Toc475004646)

[Generators 16](#_Toc475004647)

[Auto Correct Rules 17](#_Toc475004648)

[Edit Auto Correct Rule 18](#_Toc475004649)

[Ignore Record 18](#_Toc475004650)

[Append Extra Char 19](#_Toc475004651)

[Attempt ID Remote and Attempt ID Local 20](#_Toc475004652)

[Replication Status 21](#_Toc475004653)

[Force Verification 21](#_Toc475004654)

[Replication Client 22](#_Toc475004655)

[Threads 23](#_Toc475004656)

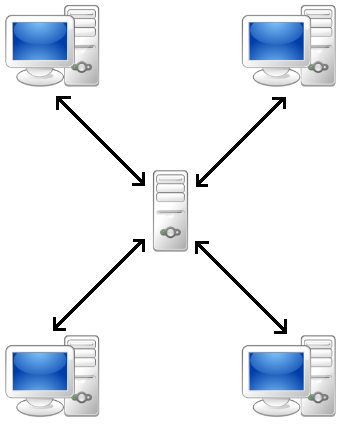
[Backup Database 24](#_Toc475004657)

[Remote Database Update 25](#_Toc475004658)

[Database Objects for Replication 26](#_Toc475004659)

# Replication

Replication enables data to be copied between the master database and multiple child databases (nodes).



Data is replicated asynchronously at user defined intervals, depending on how the replication is configured.

## Advantages of replication

There are many advantages to replication, including:

* Analytics. Analysis of information can take place on a child database, without affecting performance of the master database.
* Data distribution. Remote sites can work on local copies of data without permanent access to the master database.
* Increased availability. Should a node become unavailable due to hardware failure, clients can access an alternative node.
* Increased speed. Accessing data on a local network is generally faster than public network access.

## Disadvantages of replication

There are several disadvantages to replication, including:

* Increased disk space. Storing copies of data on different sites consumes more disk space.
* Maintaining data integrity is complex.

# Before You Begin

Before you begin using and configuring a database for replication, remote update and backup, it is essential that you first test the changes on a copy of the database you are going to setup. This can be achieved by taking a backup of your database, restoring it into a test area and making changes prior to going live.

# Steps Required for Replication

The following steps are required to add replication capability to a database. They are explained further within this document.

1. Configure Database for replication.
2. Install the replication script into the database.
3. Configure Table that will be replicated.
4. Generate and execute the Replication Add Schema.
5. Set the Generators values for the configured database.

# Steps Required to Remove Replication

The following steps are required to remove replication from a database.

1. Generate and execute the Replication Remove Schema script.
2. Uninstall the replication settings from the database.

# Change Logs

Data that is being replicated is stored within change logs; these reside within tables inside both the master and child databases. The change logs store the data required to generate the SQL statements required to replicate data between master and child databases.

Triggers are created for each operation (insert, delete or update) on the tables that are being configured for replication.

# Considerations for Replication

SieraDelta’s replication engine only works on tables that have an integer based primary key. This can be int32 or int64. Int64 provides for more flexibility and should be used wherever possible.

## Site ID

Each node requires a unique ID to identify it within the cluster. The master database is always assigned 0 (zero) as its unique ID. Child nodes require a unique ID which will identify it within the cluster.

The maximum number of sites supported is 99,999.

## Generators

SieraDelta’s replication engine supports two types of unique ID, they are:

### Unique Master ID

The unique identifier is reassigned by the master database when records are inserted. In this scenario each node has negative primary key values.

### Unique Site ID

The site ID of the node indicates the unique ID being used, generators are updated as part of the setup process as follows:

Site 1, all generators begin at a value of 100000000 for 64 bit integers or 100000 for 32 bit integers.

Site 20, all generators begin at a value of 20000000000 for 64 bit integers or 20000000 for 32 bit integers.

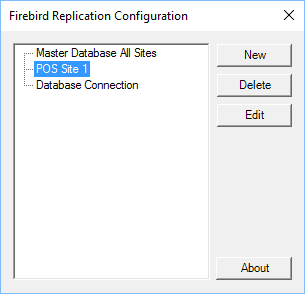
This method provides a maximum possible range of one hundred million unique identifiers for child nodes when using 64 bit integers and one million unique identifiers for 32 bit integers. These are recommended values and can be updated / changed if required depending on your database usage.

Both of the above methods can be mixed within a replicated database, for example, if you have a table which has or is expected to grow beyond one hundred million records, you can assign a negative value for the child generator and assign a positive number in the master database.

# Configuration

You can configure multiple databases on a single server for replication, each replicated database runs within its own thread.

Run the configuration utility to be shown this window



A list of all databases which are configured for Replication, remote update or automatic backup will be shown within the list.

New - Displays configuration for a new database.

Delete - Delete’s the selected database, displays a dialog for confirmation.

Edit - Edit’s the selected database.

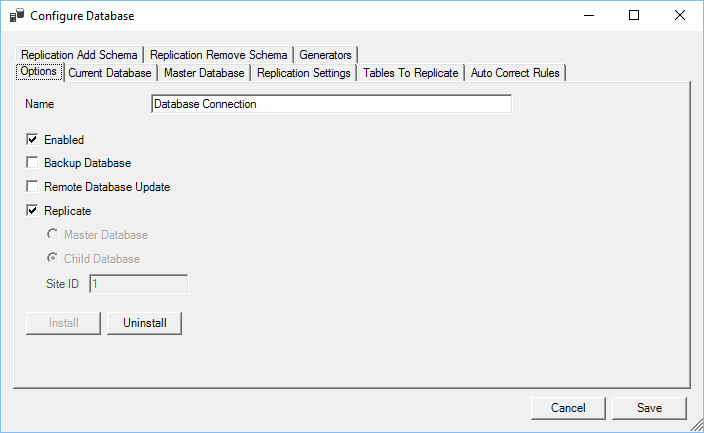
About - Displays copyright and version information, as well as information on updates.

# Configure Database

When setting up a new database, the first thing to configure is the database connection, this is available on the Current Database tab.

## Options

There are several options for initial configuration of the database.



Name - User defined name for the database connection.

Enabled - This enables the database within the service if ticked, otherwise no action will be taken.

Backup Database - If ticked then a new tab will be shown with Remote Database Update settings.

Remote Update - if ticked then the database will be eligible for

Remote Database Update and a new tab will be available for remote update settings.

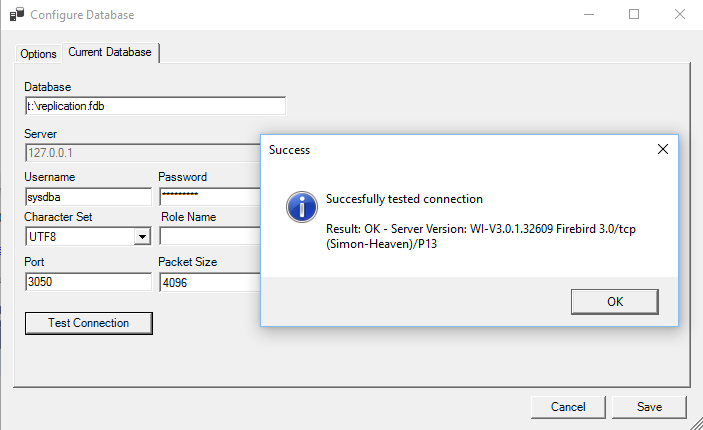
Replicate - If ticked then the database will also be setup for replication.

Install - Executes the script that will install replication into the database.

Uninstall - Executes the script that will remove replication from the database.

When initially setting up a database for replication, remote update or backup, enter the details for the current database and click “Test Connection” you will then be able to Install the script into the database. The site ID for a master database is always zero, for a child database; enter a unique (within your replication cluster) id to identify the database. Once set you cannot change the Master, Child or Site ID settings.

## Current Database



Database - The alias name or full path/name of the database on the local computer.

Server - Only the local server can be used for the current database.

Username - Database user name.

Password - Database password.

Character Set - Character set for the connection.

Role Name - Name of the database role, if required.

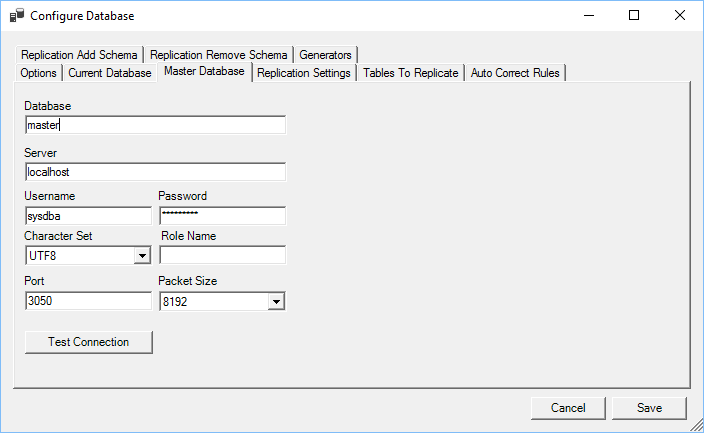
Port - Port used to connect to the Firebird server.

Packet Size - Size of packets.

Test Connection - Tests the connection to the database, when successful a dialog box will show the database version.

## Master Database

When replicating and the replication type is child, you will need to configure the master database.



Database - The alias name or full path/name of the database.

Server - The IP address of the server holding the master database.

Username - Database user name for master database.

Password - Database password for master database.

Character Set - Character set for the connection.

Role Name - Name of the database role, if required.

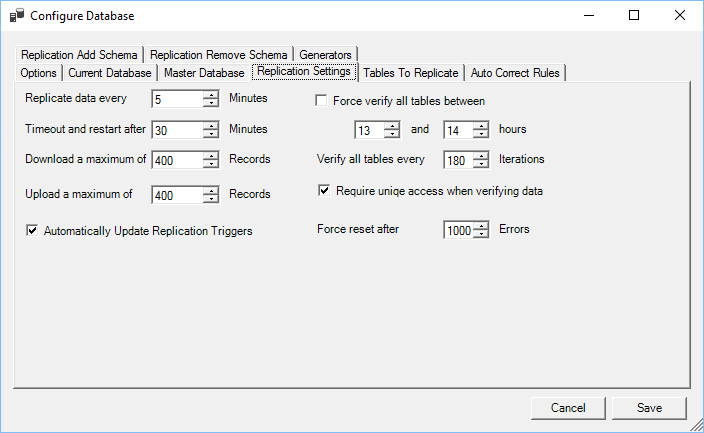
Port - Port used to connect to the Firebird server.

Packet Size - Size of packets.

Test Connection - Tests the connection to the database, when successful a dialog box will show the database version.

## Replication Settings

The replication engine provides multiple options for replication



Replicate Every - Indicates how often replication can take place; this can be between 1 and 520 minutes.

Timeout - If replication exceeds this value, then it will be restarted, any transactions processed will be committed.

Download - Indicates the maximum number of records to download from the master database during replication.

Upload - Indicates the maximum number of records to upload to the master database.

Force Verify - If ticked, then a full scan of the child and master database will be completed and all missing records replicated between the specified hours.

Unique Access - If ticked, verification of tables will not occur whilst other replication engines are connected to the master database.

Verify Tables - Verifies data after n number of replications, a value of zero disables forced verification.

Force Reset Errors - If n number of errors occur whilst replicating tables, replication is restarted.

Update Triggers - If automatically update triggers is ticked, when tables are created or altered using

Remote Database Update then the replication triggers are automatically recreated.

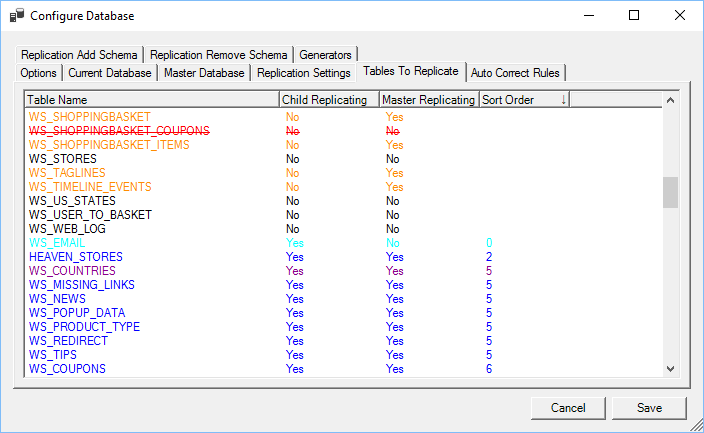
If replication is reset due to the download, upload, time out or number of errors, then it will be executed immediately without waiting for the set number of minutes to wait between replication.

Automatic updating of triggers is used in conjunction

Remote Database Update, if with remote database update is not enabled then this option will be unavailable.

## Tables to Replicate

There are multiple options when configuring tables that will be replicated.



Whilst configuring a database, the status of the table is indicated using different colours.

### Colour chart

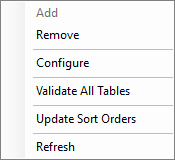
|  |  |
| --- | --- |
| Colour | Meaning |
| Black | Table is not replicated. |
| Blue | Table is replicating within the master and child database. |
| Orange | Master is replicating, child is not replicating. |
| Dark Magenta | Child table is incorrectly configured. |
| Dark Green | Table is configured differently between Master and Child. This usually means that the index names are different but both have the same index settings. |
| Red | Child table is different to Master table. |
| Red with ~~strike through~~ | Either the child or master does not have a primary key and cannot be replicated. |
| Cyan | Child is replicating, master is not replicating. |

### Sort Order

The sort order is used to determine the order in which tables are sorted when Force Verification is used. If incorrectly setup, then errors may occur when replicating data due to missing foreign keys etc.

## Menu Options

Right clicking on the tables will show a menu for manipulating tables.



Add - Add’s the table to the list of tables being replicated. When adding a table to the list a new window will be shown to configure the table.

Remove - Prevents the table from replicating by removing it from the list, there is an option when manipulating a child database to stop replicating the same table in the master database.

Configure - View Configure Table section for detailed information when configuring a table for replication.

Validate All Tables - Validates all tables and changes the colour depending on its state.

Update Sort Orders - When selected, the sort orders will be updated for all replicated tables to account for foreign keys within the database. If a table has a foreign key to another table, it will have a higher sort order. The sort order can be changed during Configure Table.

Refresh - Refreshes all replicated table data.

# Configure Table

Each eligible table can be configured for replication.

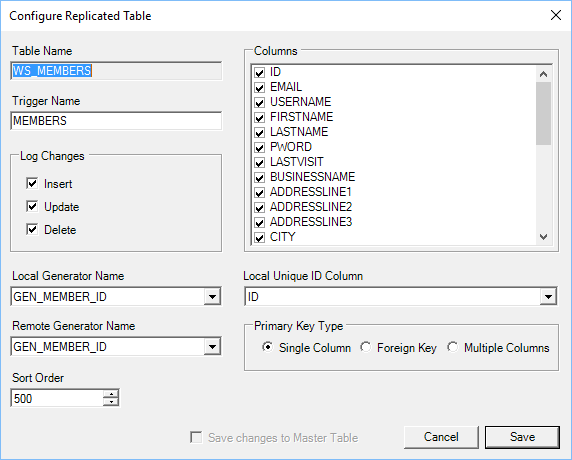


Table Name - Name of table being configured.

Trigger Name - Unique name for the trigger that will be used on the table to add data to the Change Logs. The maximum length is 18 characters and must only contain alpha numeric characters and an underscore. When triggers are created they will take the name like: REPLICATE$[TriggerName]\_[I/U/D] I, U and D refers to Insert, update and delete.

Log Changes - Select which changes are to be logged.

Columns - Columns to replicate. Any column without a tick will be excluded from the replication process.

Local Generator - Specify the name of the local generator, this value can be set to "Use Child Generated ID" however, the replication engine will use this value when determining the new generator value. See Generators section for more information.

Remote Generator - See Considerations for Replication, Generators section for more information. If this value is set then the final unique id for the record will be set when the record is replicated to the master database.

Sort Order - Predefined sort order, see Force Verification section for more details.

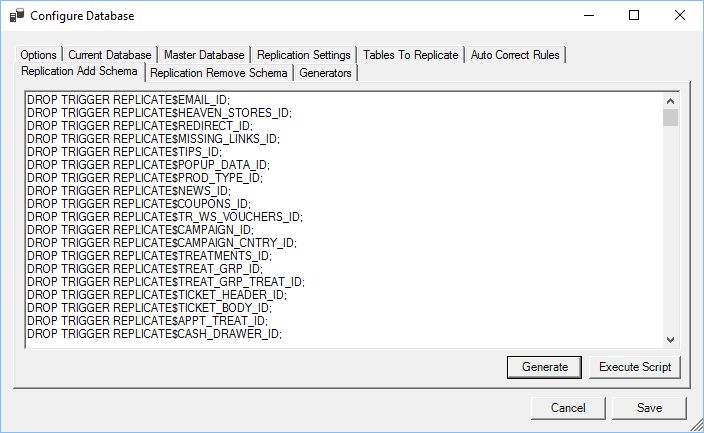
Local Unique ID - Specifies the unique id column for the table, usually the primary key.

Primary Key Type - The type of primary key, this can be set to Single Column, which indicates the primary key is a single column. Foreign Key, indicating that the primary key is also a foreign key and Multiple columns, indicating that the primary key is based on multiple columns within the table.

A check will be made of the primary key type when configuring a table, if it is deemed incorrect a dialog will be displayed informing of what type the engine thinks it should be.

# Replication Add Schema

The replication add schema tab will generate the necessary SQL to add replication to the tables.



Generate - Generates the required script, this can then be copied and executed manually.

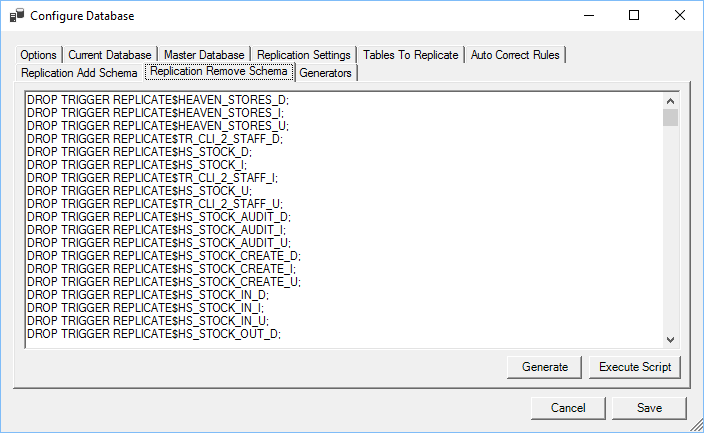
Execute Script - Generates and executes the required script to enable replication within the database.

The SQL generated does the following:

* Drop all replication triggers for the tables.
* Creates all the triggers required to replicate changes to the Change Logs.
* Adds a column to all replicating tables “REPLICATE$HASH”. This is a 64 bit int column which contains a hash of the record.
* Disables all active triggers for tables that will be replicated.
* Reset’s all Hash values for the tables that will be replicated, if it has not previously been set.
* Reactivates any triggers that were deactivated on replicated tables.

# Replication Remove Schema

The replication remove schema tab, generates the necessary SQL to remove table replication.

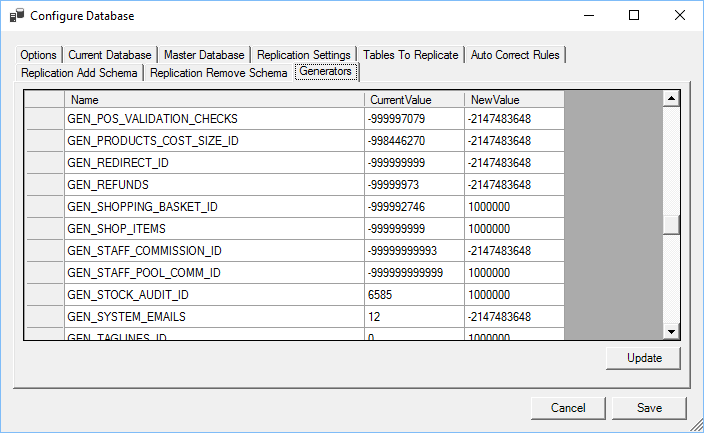


Generate - Generates the required script, this can then be copied and executed manually.

Execute Script - Generates and executes the required script to remove replication within the database.

# Generators

The generators tab provides an opportunity to reset the generator values based on the settings after configuration.



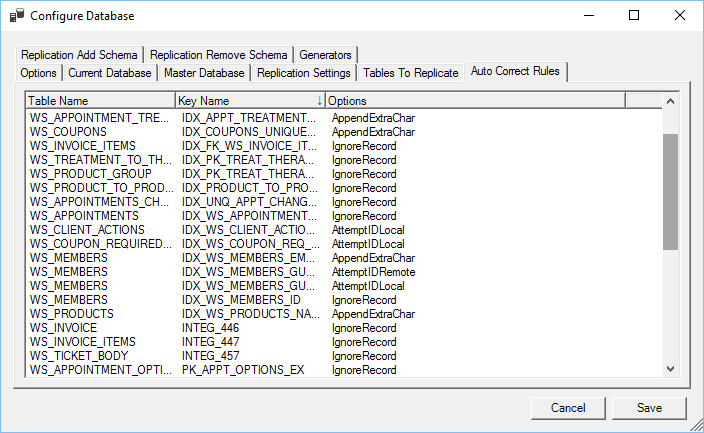
Update - Updates the generator values to the New Value specified.

Before updating values, please ensure you read the Considerations for Replication section.

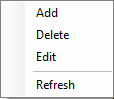
The generator value is a recommended value based on the settings when Configure Table for replication in conjunction with the site id for the child database.

# Auto Correct Rules

The replication engine can process replication record failures based on rules. In general the primary reason a record fails to replicate is due to inconsistency in a database structure between the master and child databases.



Right clicking on a record will present a menu with options for manipulating auto correct rules.



Add - Adds a new rule to the list.

Delete - Deletes the currently selected rule.

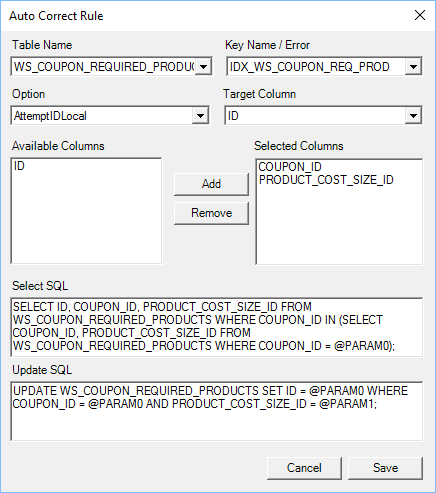
Edit - Edit’s the currently selected rule.

Refresh - Refreshes the list of rules.

When editing or creating rules you will be presented with a new window.

## Edit Auto Correct Rule

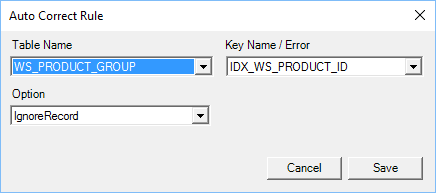
Auto correct rules can be configured for any table or key that is being replicated.



Presently there are 4 options available.

### Ignore Record

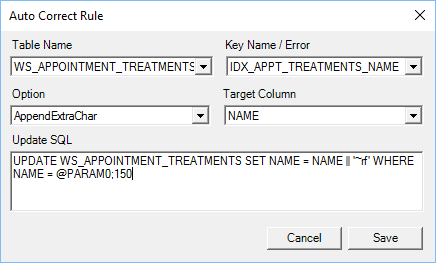
Any replication operation that fails can be ignored.



In the above example, if a replication operation fails due to index IDX\_WS\_PRODUCT\_ID on the table WS\_PRODUCT\_GROUPS will be ignored and the Replication Status changed to ignore.

### Append Extra Char

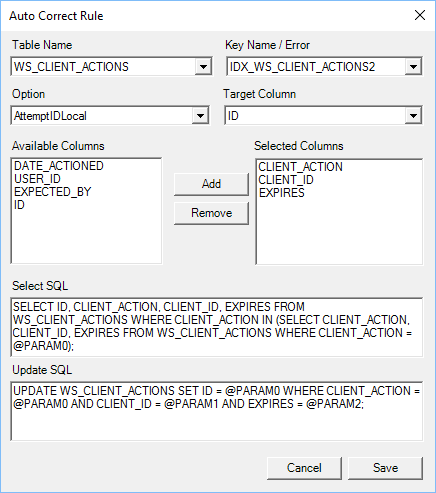
If a replication operation fails due to a conflict within a unique index on a string field, the engine can attempt to alter the text of the column causing conflicts.



In this example the record is updated to change NAME column.

### Attempt ID Remote and Attempt ID Local

If a replication operation fails due to a conflict in the indexes then the engine can attempt to update the existing record to make it comply with either the master or child.



The engine will attempt to select the matching record within the database and update the primary key of the record to match that of the master or child.

The SQL is generated automatically, but can be overridden by administrators.

# Replication Status

For child databases, the replicated state of data in the change log can be one of the following:

0. Operation not replicated.

1. Operation successfully replicated.

3. Operation has no column changes.

5. Violation of primary or unique key.

6. Operation ID is duplicated.

104. Operation has no available ID.

Records within the operation log, which are of type delete and with status of 3 should never be deleted from the table.

# Force Verification

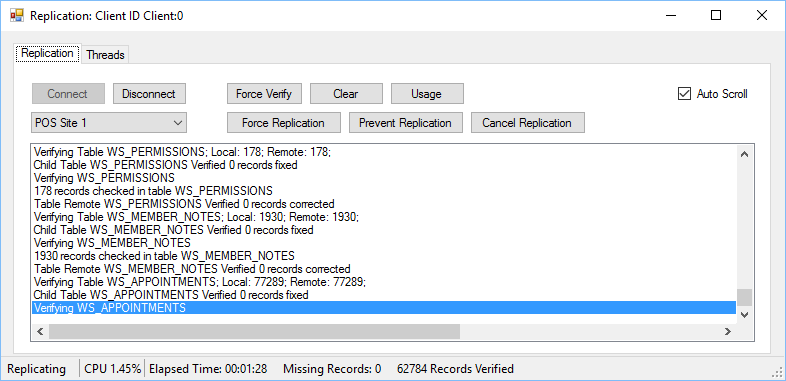
Force verification is the process of validating all records between a child and master database. This operation can, if the database is very large, be time consuming.

The replication engine will scan all records within replicated tables and verify that records match in both the child and master. Any missing records will be inserted into the database where they are missing from.

Once a table is successfully scanned, it will be included in an exclude list, if the verification process times out due to settings, a successfully scanned table will be ignored on the next iteration.

# Replication Client

SieraDelta’s replication engine contains a replication client which allows administrators to view the current status of the replication.



The drop down combo box determines which database configuration is being monitored; all commands sent to the replication engine only select the selected configuration.

Connect - Connects to the replication engine on the local computer.

Disconnect - Disconnects from the replication engine on the local computer.

Force Verify - Instructs the replication engine to Force Verification of all tables when replication is next run for the selected configuration.

Clear - Clears the list of notifications.

Usage - Updates the thread usage tab with thread information for the replication engine.

Auto Scroll - Automatically selects the last notification in the list if ticked.

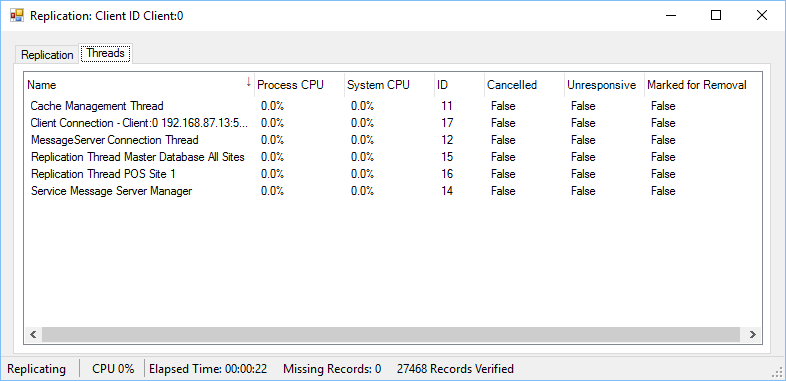
Force - Instructs the engine to begin replication for the selected configuration.

Prevent - Prevents the selected configuration from replicating.

Cancel - Cancel’s replication for the selected configuration.

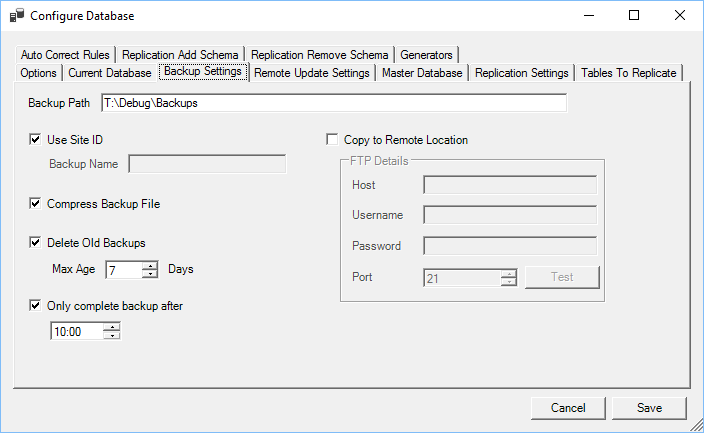
## Threads

The threads tab shows the current active threads within the replication engine and their current status.



# Backup Database

SieraDelta’s replication service includes an optional automatic daily backup module which can backup databases and optionally move a copy to a remote server via FTP.



Backup Path - The path where the file will be backed up.

Use Site ID - If true, the backup name will incorporate the site id, otherwise it will incorporate the backup name.

Backup Name - If not using a site id, the backup name will incorporate the name specified into the backup name.

Compress - If selected the backup will be compressed.

Delete Old Files - If selected then files older than Maximum Age will be deleted from the backup path.

Maximum Age - Maximum age of backups to keep.

Backup After - If set, specify the time when a backup can start.

Remote Location - If selected a copy will be sent to a remote server using FTP.

FTP Details - FTP details for a remote server where the backup will be copied to.

The name of the backup takes the following form:

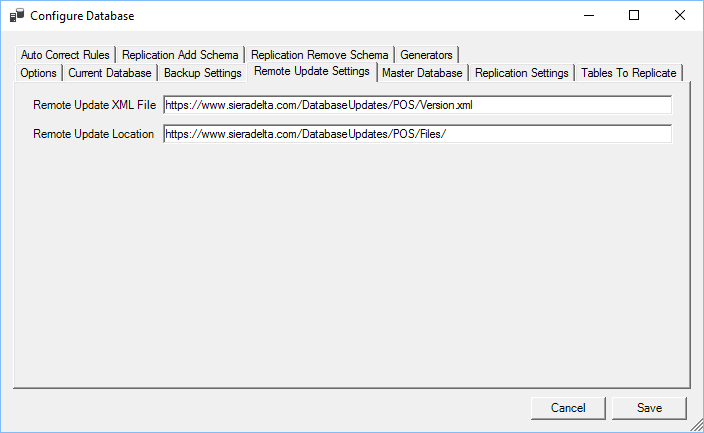
Backup-[ID or Backup Name]-[Year][Month][Day].fbk

To save system resources only one database can be backed up at a time, whilst a database is being backed up, all other databases awaiting backup will be in a queue.

If a database fails backup a further attempt is made 6 hours later.

# Remote Database Update

SieraDelta’s replication service also includes a module to optionally update a database remotely.



XML File - The full URL to an xml file which contains version information for database updates.

Update Location - The full URL to the location that update files are stored

The version XML file must follow this form:

<?xml version="1.0" encoding="utf-8"?>

<SieraDelta>

<Database>

<Version>377</Version>

</Database>

< /SieraDelta>

The update location should contain the text files which have SQL updates; the files naming convention must follow the following form:

[version].txt

So version 377 would be: 377.txt

The SQL update files are plain text files that contain DDL or DML Statements. Commit, Rollback and Set Term statements should not be included.

# Database Objects for Replication

The following objects are required within a database that is being replicated, all objects are named REPLICATE$[object name].

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Description | Master | Child |
| ID | Generator | Unique ID used within replication. | 🗶 | ✓ |
| REPLICATETABLES\_ID | Generator | Replication table ID | ✓ | ✓ |
| REMOTE\_LOG\_ID | Generator | Master Database Log ID |  | ✓ |
| OPERATIONLOG\_ID | Generator | Operation Log ID | ✓ | ✓ |
| COLUMNLOG\_ID | Generator | Column Log ID | ✓ | ✓ |
| OPERATIONLOG | Table | Table change log containing table updates, inserts and deletes. | ✓ | ✓ |
| COLUMNLOG | Table | Column change log containing column changes for each insert, update and delete. | ✓ | ✓ |
| TABLES | Table | List of tables, with options that will be replicated. | ✓ | ✓ |
| OPTIONS | Table | Replicating options and basic settings. | ✓ | ✓ |
| LOCALPKCHANGES | Table | Local primary key changes. | 🗶 | ✓ |
| AUTOCORRECTRULES | Table | Table containing auto correct rules. | ✓ | ✓ |
| REQUIRED\_SQL | Table | Contains SQL script for changes. | ✓ | 🗶 |
| BUILDWHERECLAUSE2 | Procedure | Used to create a where clause. | 🗶 | ✓ |
| BUILDWHERECLAUSE | Procedure | Used to create a where clause. | ✓ | ✓ |
| VERIFY\_RECORDS | Procedure | Used when verifying records. | ✓ | ✓ |
| QUOTESTR | Procedure | Used to determine whether a field should be quoted or not. | ✓ | ✓ |
| REPLICATEUPDATE | Procedure | Generates an update statement from change logs. | ✓ | ✓ |
| REPLICATEINSERT | Procedure | Generates an insert statement from change logs. | ✓ | ✓ |
| REPLICATEDELETE | Procedure | Generates a delete statement from change logs. | ✓ | ✓ |
| REPLICATECHANGES\_TEST | Procedure | Generates all changes. | 🗶 | ✓ |
| SPLITSTRING | Procedure | Splits a string based on values. | ✓ | ✓ |
| REPLICATECHANGES\_I | Procedure | Generates all statements for replication. |  | ✓ |
| REPLICATECHANGES | Procedure | Generates all statements for replication. | ✓ | ✓ |
| REMOTEUPDATES\_2 | Procedure | Updates transaction context. | 🗶 | ✓ |
| REMOTEUPDATES | Procedure | Executes a statement. | 🗶 | ✓ |
| OPERATIONLOG\_INSERT | Procedure | Inserts an item into the table change log. | ✓ | ✓ |
| LOCALPKCHANGES\_INSERT | Procedure | Inserts a record into the local primary key change log. | 🗶 | ✓ |
| GENERATEINSERT | Procedure | Generates an insert statement from change logs. | 🗶 | ✓ |
| FIND\_MISSING\_RECORDS | Procedure | Searches for missing records within a table. | ✓ | ✓ |
| COLUMNLOG\_INSERT | Procedure | Inserts changes for column change log. | ✓ | ✓ |
| AUTOFIXRECORD | Procedure | Attempts to fix the primary key in column log changes if the record fails to replicate due to violation of foreign key. | 🗶 | ✓ |
| REMOTECOUNT | Procedure | Retrieves number of available updates from the master database. | ✓ | 🗶 |
| RUNSQL | Procedure | Executes SQL changing the primary key value if required. | ✓ | 🗶 |
| MISSINGRECORD | Procedure |  | ✓ | 🗶 |
| REPLICATEUPDATE | Procedure |  | ✓ | 🗶 |
| REPLICATEINSERT | Procedure |  | ✓ | 🗶 |
| REPLICATEDELETE | Procedure |  | ✓ | 🗶 |
| CREATE\_MISSING\_RECORD | Procedure |  | ✓ | 🗶 |