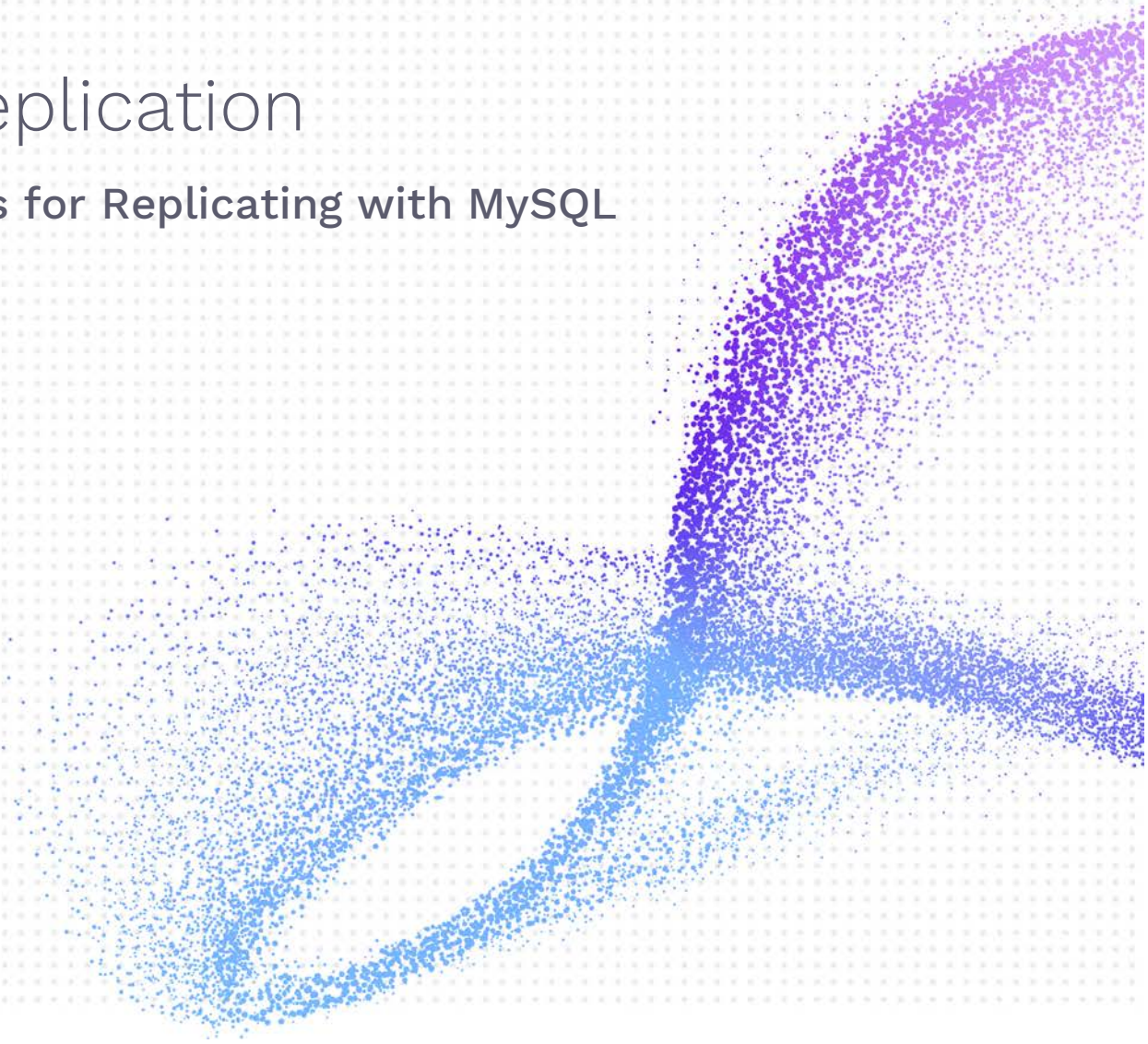
Syniti

Data Replication

Setup Notes for Replicating with MySQL

Version 9.8.2



Syniti

Syniti Data Replication

Table of Contents

Syniti Data Replication

These notes provide essential information for setting up replications using **MySQL** as a source database for one-way mirroring and synchronization**.** The setup process for a refresh replication can usually be completed using the Syniti Data Replication wizards without additional documentation because it does not involve access to the MySQL logs.

This guide describes the setup process using either the Log Reader or Log Server Agent options for one-way mirroring and synchronization when replicating data from a MySQL database. For mirroring and synchronization replications using MySQL as a source, Syniti Data Replication offers several approaches:

- **Log Reader**: Queries the MySQL Log directly
- **Log Server Agent**: Uses a Windows service and a Log Server component to query the MySQL Log for increased performance when dealing with large amounts of data.
- **Triggers**: Uses Syniti DR triggers installed on the MySQL database to log changes (Ask the technical support team at http://support.hitsw.com to see if this approach would work in your application and to obtain the separate document *Setup Notes for Trigger-based Transactional Replications*.).

For complete details on the setup process, check the *Syniti Data Replication User Guide* available from the Management Center **Help** menu.

Connection Type

MySQL .NET Data Provider available for download from MySQL

Note: Trigger-based transactional replications are supported for v. 4.1.5 and later; log- and trigger-based transactional replications are supported for v. 5.1.31 and later

Assembly: MySql.Data (file name: MySql.Data.dll)

Sample path: C:\MySQL\Connector-net-5.0.3\Driver\bin\net2.0\release\MySql.Data.dll

Additional Files Required for MySQL Log-based Replications

If you are planning to set up replications using the MySQL binary log, contact Technical support to obtain an additional component (DBMySqlUtil.zip). Be sure to state whether you are using the 32-bit or 64-bit version of Syniti Data Replication.

When you receive the zip file, unzip it and place the contents in the Syniti Data Replication installation folder (usually C:\Program Files\Syniti\Data Replication)

User Permissions for MySQL

When setting up replications that use MySQL as either a source or target database, you need to be sure that the user ID used for making connections to the database has sufficient privileges to complete all the operations required for Syniti Data Replication to perform a replication.

This section is organized by the type of replication you want to perform. It describes in detail all the user authorities that will be required during the setup and execution of replications.

Refresh with MySQL as Either Source or Target Database

1. AUTHORITY TO CONNECT TO DATABASE
   This should already be granted when the user is created. However, here is the syntax, just in case. The command below actually creates a new user, granting basic privileges. Privileges must be assigned to both the localhost and '%' to give access remotely from any client machine (see MySQL documentation at

https://dev.mysql.com/doc/).The FLUSH PRIVILEGES command make the changes immediately available without restarting the MySQL service.

```
USE <SCHEMA>;
GRANT USAGE ON *.* TO '<UID>'@'localhost' IDENTIFIED BY '<PWD>';
GRANT USAGE ON *.* TO '<UID>'@'%' IDENTIFIED BY '<PWD>';
FLUSH PRIVILEGES;
```

Example where sdruser is the user ID and mysql is the schema:

```
USE mysql;
GRANT USAGE ON *.* TO 'sdruser'@'localhost' IDENTIFIED BY 'sdruser';
GRANT USAGE ON *.* TO 'sdruser'@'%' IDENTIFIED BY 'sdruser';
FLUSH PRIVILEGES;
```

2. AUTHORITY TO SELECT CATALOG
   No additional permissions needed.

3. AUTHORITY TO SELECT TABLES
   Needed for schemas other than the one for which CONNECT_DB permission has been granted (the user has already read authority.)

   ```
   GRANT SELECT ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
   GRANT SELECT ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
   FLUSH PRIVILEGES;
   ```

   Example where sdruser is the user ID and mysql is the schema:

   ```
   GRANT SELECT ON mysql.Table1 TO 'sdruser'@'localhost';
   GRANT SELECT ON mysql.Table1 TO 'sdruser'@'%';
   FLUSH PRIVILEGES;
   ```

4. AUTHORITY TO CREATE TABLES
   To create a target table in the Management Center (as part of the Create Table Wizard), Syniti Data Replication requires permissions to modify the database schema.

   MySQL allows you to grant privileges even on database objects that do not exist. In such cases, the privileges to be granted must include the **CREATE** privilege. This behavior is designed to enable the database administrator to prepare user accounts and privileges for database objects that are to be created at a later time (see `http://dev.mysql.com/doc/refman/5.0/en/grant.html`).

   ```
   GRANT CREATE ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
   GRANT CREATE ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
   FLUSH PRIVILEGES;
   ```

Alternatively, you can run the command using the '*' wildcard:

```
GRANT CREATE ON <SCHEMA>.* TO '<UID>'@'localhost';
GRANT CREATE ON <SCHEMA>.* TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

Example where sdruser is the user ID and mysql is the schema:

```
GRANT CREATE ON mysql.Table1 TO 'sdruser'@'localhost';
GRANT CREATE ON mysql.Table1.* TO 'sdruser'@'%';
FLUSH PRIVILEGES;
```

5. AUTHORITY TO UPDATE TABLES

   To create a target table in the Management Center (as part of the Create Table Wizard), Syniti Data Replication requires write permissions to the database. The following command is needed for schemas other than the one on which the CONNECT_DB permission has been granted (where the user has already write authority.)

```
GRANT INSERT, UPDATE, DELETE ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
GRANT INSERT, UPDATE, DELETE ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

   Example where sdruser is the user ID and mysql is the schema:

```
GRANT INSERT, UPDATE, DELETE ON mysql.Table1 TO 'sdruser'@'localhost';
GRANT INSERT, UPDATE, DELETE ON mysql.Table1 TO 'sdruser'@'%';
FLUSH PRIVILEGES;
```

6. AUTHORITY TO DROP TABLES, ALTER TABLES (optional)

   The use of these commands from within Syniti Data Replication is entirely optional (i.e. not necessary for running a refresh replication.) They are used if you choose to remove a table from MySQL or change the table via the Management Center SQL Query tab. This syntax can be used for schemas other than the one for which the CONNECT_DB permission has been granted (where the user has already drop table authority.)

```
GRANT DROP ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
GRANT DROP ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

   Example where sdruser is the user ID and mysql is the schema:

```
GRANT DROP ON mysql.Table1 TO 'sdruser'@'localhost';
GRANT DROP ON mysql.Table1 TO 'sdruser'@'%';
FLUSH PRIVILEGES;
```

Transactional Replications/Initial Refresh with MySQL as Either Source or Target Database

This section includes information for mirroring where MySQL is the data source, and synchronization where MySQL can be either the “source” or “target” data source.

1. AUTHORITY TO CONNECT TO DATABASE
   This should already be granted when the user is created. However, here is the syntax, just in case. The command below actually creates a new user, granting basic privileges. Privileges must be assigned to both the localhost and ‘%’ to give access remotely from any client machine (see MySQL documentation at https://dev.mysql.com/doc/).The FLUSH PRIVILEGES command makes the changes immediately available without restarting the MySQL service.

   ```
   USE <SCHEMA>;
   GRANT USAGE ON *.* TO '<UID>'@'localhost' IDENTIFIED BY '<PWD>';
   GRANT USAGE ON *.* TO '<UID>'@'%' IDENTIFIED BY '<PWD>';
   FLUSH PRIVILEGES;
   ```

   Example where sdruser is the user ID and mysql is the schema:

   ```
   USE mysql;
   GRANT USAGE ON *.* TO 'sdruser'@'localhost' IDENTIFIED BY 'sdruser';
   GRANT USAGE ON *.* TO 'sdruser'@'%' IDENTIFIED BY 'sdruser';
   FLUSH PRIVILEGES;
   ```

2. AUTHORITY TO SELECT CATALOG
   No additional permissions needed.

3. AUTHORITY TO SELECT TABLES
   Needed for schemas other than the one for which CONNECT_DB permission has been granted (the user has already read authority.)

   ```
   GRANT SELECT ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
   GRANT SELECT ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
   FLUSH PRIVILEGES;
   ```

   Example where sdruser is the user ID and mysql is the schema:

   ```
   GRANT SELECT ON mysql.Table1 TO 'sdruser'@'localhost';
   GRANT SELECT ON mysql.Table1 TO 'sdruser'@'%';
   FLUSH PRIVILEGES;
   ```

4. AUTHORITY TO CREATE TABLES
   To create a target table in the Management Center (as part of the Create Table Wizard), Syniti Data Replication requires permissions to modify the database schema.

MySQL allows you to grant privileges even on database objects that do not exist. In such cases, the privileges to be granted must include the **CREATE** privilege. This behavior is designed to enable the database administrator to prepare user accounts and privileges for database objects that are to be created at a later time (see http://dev.mysql.com/doc/refman/5.0/en/grant.html).

```
GRANT CREATE ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
GRANT CREATE ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

Alternatively, you can run the command using the '*' wildcard:

```
GRANT CREATE ON <SCHEMA>.* TO '<UID>'@'localhost';
GRANT CREATE ON <SCHEMA>.* TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

Example where sdruser is the user ID and mysql is the schema:

```
GRANT CREATE ON mysql.Table1 TO 'sdruser'@'localhost';
GRANT CREATE ON mysql.Table1.* TO 'sdruser'@'%';
FLUSH PRIVILEGES;
```

5. AUTHORITY TO UPDATE TABLES

   To create a target table in the Management Center (as part of the Create Table Wizard), Syniti Data Replication requires write permissions to the database. The following command is needed for schemas other than the one on which the CONNECT_DB permission has been granted (where the user has already write authority.)

   ```
   GRANT INSERT, UPDATE, DELETE ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
   GRANT INSERT, UPDATE, DELETE ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
   FLUSH PRIVILEGES;
   ```

   Example where sdruser is the user ID and mysql is the schema:

   ```
   GRANT INSERT, UPDATE, DELETE ON mysql.Table1 TO 'sdruser'@'localhost';
   GRANT INSERT, UPDATE, DELETE ON mysql.Table1 TO 'sdruser'@'%';
   FLUSH PRIVILEGES;
   ```

6. AUTHORITY TO DROP TABLES, ALTER TABLES (optional)

   The use of these commands from within Syniti Data Replication is entirely optional (i.e. not necessary for running a refresh replication.) They are used if you choose to remove a table from MySQL or change the table via the Management Center SQL Query tab. This syntax can be used for schemas other than the one for which the CONNECT_DB permission has been granted (where the user has already drop table authority.)

   ```
   GRANT DROP ON <SCHEMA>.<TABLE> TO '<UID>'@'localhost';
   GRANT DROP ON <SCHEMA>.<TABLE> TO '<UID>'@'%';
   ```

```
FLUSH PRIVILEGES;
```

Example where sdruser is the user ID and mysql is the schema:

```
GRANT DROP ON mysql.Table1 TO 'sdruser'@'localhost';
GRANT DROP ON mysql.Table1 TO 'sdruser'@'%';
FLUSH PRIVILEGES;
```

7. AUTHORITY TO SET UP TRANSACTIONAL REPLICATIONS and ACCESS LOG

   For transactional replications using **triggers**, the following authorities are needed: read-write access, create and drop access to _dbm__trg_objs and to _dbm__masterlog, as well as read-write access to the single log files. Authority is needed for schemas other than the one for which CONNECT_DB permission has been granted (where the user has already all authorities). Log files do not have a fixed name but are generated on the fly. Therefore, the authority needs to be granted on a schema level. Notice that the last GRANT command covers also the GRANT assigned to the _dbm__trg_objs and to _dbm__masterlog tables. <SCHEMA> in all cases below should be the schema where the master table is defined.

```
GRANT SELECT, INSERT, UPDATE, DELETE ON <SCHEMA>._dbm__trg_objs TO '<UID>'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON <SCHEMA>._dbm__trg_objs TO '<UID>'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON <SCHEMA>._dbm__masterlog TO '<UID>'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON <SCHEMA>._dbm__masterlog TO '<UID>'@'%';
GRANT SELECT, INSERT, UPDATE, DELETE ON <SCHEMA>.* TO '<UID>'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON <SCHEMA>.* TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

   For MySQL versions 5.1.6 and above, trigger-based replications, also execute the following :

```
GRANT TRIGGER ON <SCHEMA>.* TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

   For transactional replications using the **MySQL binary log**, you need to grant REPLICATION SLAVE privilege to the Syniti Data Replication MySQL user ID that will connect to the database.

```
GRANT SELECT, PROCESS, REPLICATION CLIENT, REPLICATION SLAVE, RELOAD ON *.* TO '<UID>'@'%';
FLUSH PRIVILEGES;
```

   Example where user ID that is used to replicate data is sdruser:

```
GRANT SELECT, PROCESS, REPLICATION CLIENT, REPLICATION SLAVE, RELOAD ON *.* TO
sdruser@'%';
FLUSH PRIVILEGES;
```

   Note: If using MySQL versions before 5.5, you also need to grant the following privileges: FILE, SUPER

```
GRANT SELECT, PROCESS, FILE, SUPER, REPLICATION CLIENT, REPLICATION SLAVE, RELOAD ON
```

```
*.* TO sdruser@'%';
FLUSH PRIVILEGES;
```

MySQL System Settings

If you are using MySQL version 5.1.31 or above as your source database for mirroring via the MySQL binary log, you need to configure MySQL so that Syniti Data Replication can read the database's binary log.

Before Syniti Data Replication can read the binary log, row-based logging must be enabled for the database. To enable row-based logging, properties must be set in the MySQL Server Instance Configuration file.

1. Find the file "my.ini" which can be found in the directory where MySQL is installed.
   This file is the "MySQL Server Instance Configuration File".
2. Open the file my.ini.
3. Scroll to the end of the file and add the following properties:
   #For Syniti Data Replication replication
   log-bin=binlog
   binlog-format=ROW
   server-id=111
   - log-bin=binlog
     In case users want to locate them, the binlog files are found in directory specified by "datadir" in "my.ini"
   - binlog-format=ROW
     You can force the replication format by starting the MySQL server with --binlog-format=type. When set, all replication slaves connecting to the server will read the events according to this setting. ROW causes replication to be row-based.
   - server-id=<*unique_value*>
     For each server participating in replication, you should pick a positive integer in the range from 1 to 232 – 1, and each ID must be different from every other ID in use by any other replication master or slave. For example, you could set the value as 123:
     server-id=123
4. Save the my.ini file.
5. Restart the server.

Add Source Connection Wizard

Select Provider Screen

Database

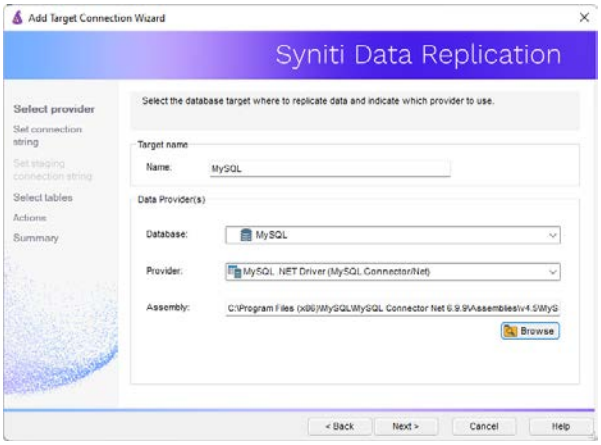
Select MySQL from the drop-down list.

Provider

The value MySQL .NET Driver is provided automatically. You do not need to change this value.

Assembly

The pathname to the .NET Assembly MySql.Data.dll. If the value is not available, Syniti Data Replication displays a message when you continue in the Source Connection wizard, allowing you to go back and type in the path. Find out the location of the assembly in your environment by searching for the file name, then enter the path and the assembly file name.



Enable Transactional Replication Wizard

For transactional replications (mirroring and synchronization), use the Enable Transactional Replication wizard after setting up a source connection. The following field(s) require specific information for SQL Server.

For transactional replications (mirroring and synchronization), use the Enable Transactional Replication wizard after setting up a source connection. The following field(s) require specific information for MySQL.

Log Type Screen

Select whether you plan to perform replications using the Log Reader (default) or Log Server Agent.

Log Settings Screen

Case Sensitive Table Names

Check this option to manage databases with case-sensitive table names.