Informatica® Data Replication
9.7.0

User Guide

Informatica Data Replication User Guide
9.7.0
July 2017

Publication Date: 2018-07-03

Table of Contents

Preface

This *Informatica Data Replication User Guide* describes how to use the Informatica® Data Replication product and user interfaces to perform transactional data replication and initial target materialization.

The guide covers the following topics:

- Data Replication architecture, usage scenarios, and deployment topologies.
- Preparing source and target databases for data replication.
- Replication considerations for specific sources and targets.
- Fundamentals of Data Replication processing, including Extractor and Applier processing, DDL replication, recovery and checkpoint processing, character set conversion, and datatype conversion.
- Using InitialSync for target materialization.
- Configuring the Server Manager.
- Using the Data Replication Console to configure, administer, schedule, monitor, and manage data replication jobs.
- Running replication jobs and tasks.
- Using the Replication Configuration Command Line Interface.
- Command-line parameters for the Extractor, Applier, Server Manager, and InitialSync.
- Data Replication glossary.

This guide is intended for system administrators, DBAs, and those who are responsible for configuring and administering data replication jobs.

For information about installing and upgrading Data Replication and supported operating system and database versions, see the *Data Replication Installation and Upgrade Guide*.

Informatica Resources

Informatica Network

Informatica Network hosts Informatica Global Customer Support, the Informatica Knowledge Base, and other product resources. To access Informatica Network, visit https://network.informatica.com.

As a member, you can:

- Access all of your Informatica resources in one place.
- Search the Knowledge Base for product resources, including documentation, FAQs, and best practices.

- View product availability information.
- Review your support cases.
- Find your local Informatica User Group Network and collaborate with your peers.

Informatica Knowledge Base

Use the Informatica Knowledge Base to search Informatica Network for product resources such as documentation, how-to articles, best practices, and PAMs.

To access the Knowledge Base, visit https://kb.informatica.com. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at KB_Feedback@informatica.com.

Informatica Documentation

To get the latest documentation for your product, browse the Informatica Knowledge Base at https://kb.informatica.com/_layouts/ProductDocumentation/Page/ProductDocumentSearch.aspx.

If you have questions, comments, or ideas about this documentation, contact the Informatica Documentation team through email at infa_documentation@informatica.com.

Informatica Product Availability Matrixes

Product Availability Matrixes (PAMs) indicate the versions of operating systems, databases, and other types of data sources and targets that a product release supports. If you are an Informatica Network member, you can access PAMs at https://network.informatica.com/community/informatica-network/product-availability-matrices.

Informatica Velocity

Informatica Velocity is a collection of tips and best practices developed by Informatica Professional Services. Developed from the real-world experience of hundreds of data management projects, Informatica Velocity represents the collective knowledge of our consultants who have worked with organizations from around the world to plan, develop, deploy, and maintain successful data management solutions.

If you are an Informatica Network member, you can access Informatica Velocity resources at http://velocity.informatica.com.

If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.

Informatica Marketplace

The Informatica Marketplace is a forum where you can find solutions that augment, extend, or enhance your Informatica implementations. By leveraging any of the hundreds of solutions from Informatica developers and partners, you can improve your productivity and speed up time to implementation on your projects. You can access Informatica Marketplace at https://marketplace.informatica.com.

Informatica Global Customer Support

You can contact a Global Support Center by telephone or through Online Support on Informatica Network.

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:
http://www.informatica.com/us/services-and-training/support-services/global-support-centers.

If you are an Informatica Network member, you can use Online Support at http://network.informatica.com.

CHAPTER 1

Data Replication Overview

This chapter includes the following topics:

Product Overview

Informatica Data Replication is a transactional data replication solution that is designed to meet the needs of enterprise-wide, mission-critical systems that need data in real time.

Data Replication replicates transactional data between heterogeneous databases and platforms while maintaining transactional integrity and data consistency. Data Replication performs low-latency batched replication or continuous replication of SQL changes and metadata from source databases to target databases over a LAN or WAN. Data Replication is nonintrusive to the underlying database systems and user applications because it uses log-based capture techniques. Data Replication also uses checkpoint processing to provide for data recovery if failures occur.

Use Data Replication to distribute, migrate, and replicate data across your environment. Data Replication is scalable. It can help you migrate data with minimal downtime and perform auditing and operational reporting functions.

Data Replication includes an efficient bulk data load tool, called InitialSync. You can use InitialSync to do an initial load of data from a source database to a target database before starting change data replication the first time.

Alternatively, if you have a license for Informatica Fast Clone, you can use Fast Clone to perform a high-speed initial load of Oracle data to target systems. Fast Clone works with Oracle sources only. It loads data to most of the same targets that Informatica Data Replication supports and also has a data streaming capability for Greenplum and Teradata targets.

Data Replication Usage Scenarios

You can use Data Replication in many scenarios where you need to replicate table inserts, updates, and deletes and metadata between homogeneous database systems or heterogeneous database systems.

The following list describes some common Data Replication usage scenarios:

Offload query processing to a less busy system

If you need to query tables in a production database that is on a source system with resource restrictions, use Data Replication to replicate data in near real time to another less expensive system. You can then run read-only queries against the data on the less-expensive system. By offloading query activity, you can reduce the load on a busy source production system to meet its high availability requirements. For example, you could replicate Oracle data to a PostgreSQL server and run queries against the data on the PostgreSQL system. Data Replication is also useful in many other load-balancing scenarios.

Provide up-to-date data for operational reporting

If you need current analytics from transactional applications on a production system to make tactical business decisions or provide business transparency, use Data Replication to replicate data in near real time to an operational data store (ODS) on another system. You can then use the ODS to get detailed information about your customers, business operations, and business trends.

Generate audit records for OLTP systems

With Data Replication, you can perform near-real-time auditing of a database, which is critical for compliance with regulatory requirements. Audit information includes the before and after images of data, the time at which the data was changed, and the SQL operation type. You can use audit information to identify patterns of user activity or develop an alert tracking system for fraud detection.

Migrate databases with minimal down time

Use Data Replication, optionally in combination with Informatica Fast Clone, as a fast database migration solution. Data Replication works in a heterogeneous environment which enables the migration of data from various databases running on different hardware and operating systems. Minimal downtime results in minimal disruption to mission-critical applications and production systems. If necessary, you can replicate data back to an older system for fallback purposes.

Data Replication Sources and Targets

Data Replication supports the following relational databases as sources:

- DB2 for Linux, UNIX, and Windows
- Microsoft SQL Server
- MySQL
- Oracle

Data Replication supports the following relational databases, file types, and data warehouse appliances as targets:

- Amazon Redshift
- Apache Kafka
- Cloudera

- DB2 for Linux, UNIX, and Windows
- Flat files
- Greenplum
- Hortonworks
- Microsoft SQL Server
- MySQL
- Netezza
- Oracle
- PostgreSQL
- Teradata
- Vertica

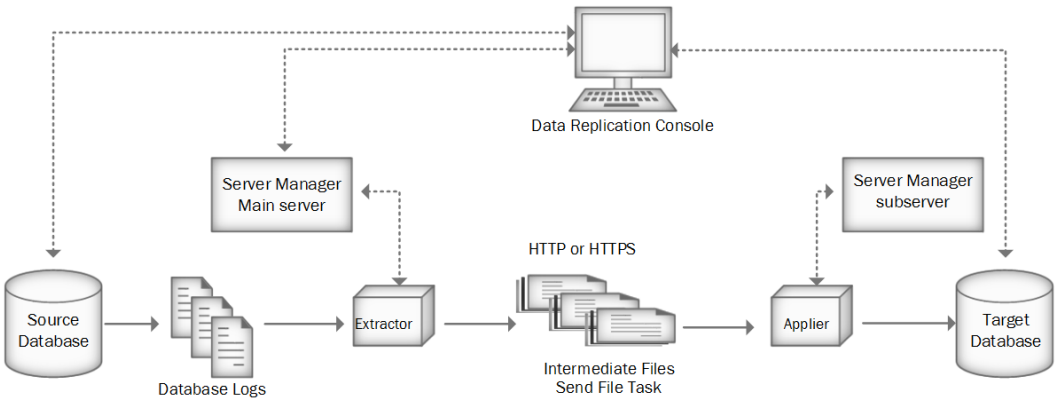
Data Replication Architecture and Components

Informatica Data Replication is composed of the following components, which run as separate executables:

- Informatica Data Replication Console, also referred to as the Data Replication Console
- Server Manager
- InitialSync
- Extractor
- Applier
- Server Manager Command Line Interface
- Replication Configuration Command Line Interface

You can run the Extractor and Server Manager on any machine that can access the source database logs. In this case, you would not install Data Replication on the source system. Also, you can run the Applier on any machine that can access the target database by using the target database connectivity client or ODBC driver.

The following image shows an example Data Replication installation:



Related Topics:

- "Alternative Deployment Topologies" on page 22

Data Replication Console

The Informatica Data Replication Console is a graphical user interface for configuring and administering data replication on Windows operating systems.

Use the Console to perform the following tasks:

- Create a replication configuration
- Configure a Server Manager
- Schedule and run a replication job
- Monitor a replication job
- Deploy a replication configuration to another replication environment
- Manage replication configurations, users, and notifications

Note: A Server Manager must be running to perform all of these tasks except configuring a new Server Manager instance.

When creating a replication configuration, you define the source database and tables, target database and tables, mappings between the source and target tables and columns, and other settings. The Console includes a schema generation option that generates compatible target table definitions based on selected source table definitions. The Data Replication Console leverages JDBC connectivity to extract the metadata that is required for mappings and execution of generated DDL.

For each replication configuration you create, Data Replication generates the following three SQLite database files that are stored in the *DataReplication_installation*/configs directory:

- *configuration_name*.db. This database file stores the table, column, and mapping definitions as well as Extractor processing information for the configuration. The Extractor uses this configuration database file to write the list of processed database logs and checkpoint information.
- *configuration_name*_buf.db. This database file contains only the processing information for the sendFile task, including the list of intermediate files that have been transferred to the Applier.
- *configuration_name*_loader.db. This database file contains processing information for Applier tasks, such as intermediate file statuses, a list of committed SCNs, and statistics.

If a replication configuration uses different servers for the source and target, a copy of the *configuration_name*.db file is stored in the *DataReplication_installation*/configs directory on both server machines. The *configuration_name*_buf.db file is stored only on the source database machine.

Informatica recommends that you use the Data Replication Console to manage configurations and replication jobs. You can run the Data Replication Console standalone on a 64-bit Windows computer system as long as the Console has JDBC connectivity to the source and target systems.

Server Manager

The Server Manager acts as the central gateway for configuration, management, and monitoring of all replication jobs and tasks. Also, the Server Manager enables data movement from a source database to a target database over a LAN or WAN. Its flexible and advanced configuration management and monitoring capabilities enable you to stream data to multiple target systems while maintaining data consistency.

In a distributed Data Replication topology, you can run the Server Manager on multiple machines. Typically, you run a Server Manager instance on every machine where an InitialSync, Extractor, or Applier component

runs. Define a Main server instance for any one of the machines in the topology and then add a Server Manager subserver to the Main server for each of the other machines. The Main server and its associated subservers comprise a replication server group.

For each Server Manager instance, Data Replication generates the following three SQLite database files to store information about tasks, schedules, configurations, connections, and statistical information:

- SM.db3
- SM_stat.db3
- lockDB.db3

These database files are stored in the *DataReplication_installation* directory.

Extractor

The Extractor reads the transaction logs of the source database system, generates intermediate files, and extracts data changes such as inserts, updates, and deletes for the selected source tables, as defined in the configuration file. The Extractor can also capture some DDL changes. The Extractor writes the changes to intermediate files.

Note: For MySQL sources, the Extractor uses the mysqlbinlog utility to read change events from the binary log.

Usually, you run the Extractor on the source system. However, you can run the Extractor on another system provided that the Extractor can access the transaction logs. For example, an Extractor that runs on Windows can parse transaction logs on an HP-UX source system. Although the Extractor parses source data efficiently, extraction performance also depends on the hardware and I/O speed for accessing transaction logs.

Applier

The Applier reads only committed changes from the generated intermediate files and applies these changes to the target. The Applier combines all changes from committed transactions since the last apply operation and then applies these changes to the target in one transaction. After the data is committed, it is purged from the intermediate files.

Usually, you run the Applier on the target system. However, you can run the Applier on another system provided that it can access the target database by using the target database connectivity client or ODBC.

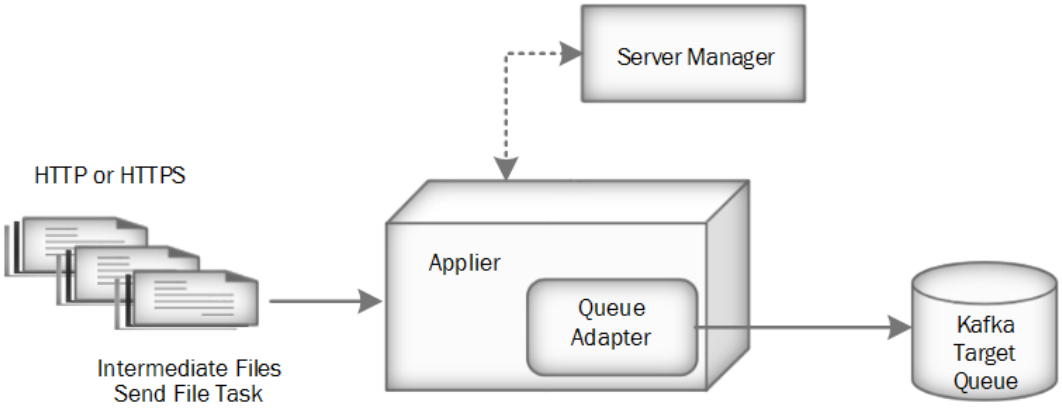
If an apply operation fails, the entire batch of changes is rolled back on the target database. If you restart the Applier, or if you shut down the database and then restart the database, the Applier can accumulate the changes and restart from the last checkpoint.

The Applier can apply source DDL changes to targets for which DDL replication is supported. DDL replication is not supported for Apache Kafka targets.

Apache Kafka Apply Process

To send change data to a Apache Kafka message queue, the Applier uses the Informatica Queue Adapter.

The following image shows the basic architecture of the Data Replication Applier for Kafka targets:



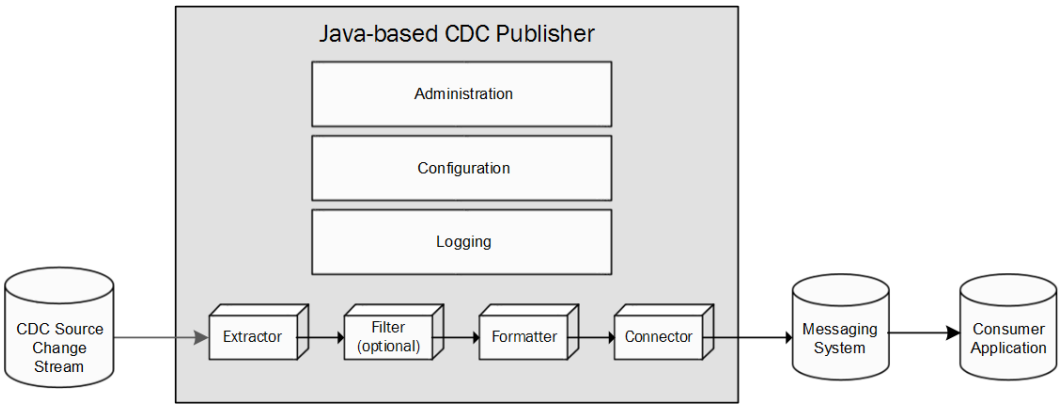
CDC Publisher

The Informatica CDC Publisher is a Java-based tool that is used to stream change data to a target messaging system such as Apache Kafka.

The CDC Publisher contains the following components that move data:

- The CDC Publisher *Extractor* consumes a stream of change data from the source. The incoming data records include schema information, row-based data changes, and transactional boundary metadata. The Extractor performs the following functions:
  - Assigns a sequence ID that is both repeatable and increasing to each change data record.
  - Interacts with the component that supplies the streamed data.
  - Ignores records that are older than the current restart point.
  - Verifies that data is in an expected format.
  - Places the results on an outbound queue for Formatter processing.
- The *Filter* component optionally filters the extracted change data based on lists of source objects to include or exclude that you specify.
- The *Formatter* receives change data from the CDC Publisher Extractor, formats the data based on the generated Avro schema of the selected format (flat, nested, or generic) for inclusion in messages, and sends the formatted messages to the Connector.
- The *Connector* reads the formatted messages from the Formatter and connects to the target messaging system to apply the messages. The Connector applies the message data in a consistent, ordered, and recoverable manner.

The following image shows the basic architecture of the Java-based CDC Publisher:



InitialSync

InitialSync materializes target tables from source tables. You can run InitialSync from the Data Replication Console, command line, or scripts. Use it to perform an initial load of data prior to starting change data replication the first time.

By default, InitialSync runs on the source Server Manager instance. However, you can create another InitialSync task and configure it to run on any Server Manager instance in the replication topology.

Intermediate Files

The Extractor generates intermediate files on disk to store captured change data. An intermediate file is composed of a data file (.dat) and a transaction file (.trn).

The Server Manager instances on the source and target coordinate to transfer the intermediate files to the Applier. The Applier parses the files and applies the committed changes to the target database. After the Applier sends a confirmation message to the Server Manager instance on the target, the Server Manager purges the intermediate files for which all changes were processed and retains only the intermediate files that contain uncommitted transactional changes.

Server Manager Command Line Interface

The Server Manager Command Line Interface (CLI) is a Server Manager command line utility that you can use to manage Data Replication configurations, schedules, and tasks. For more information, see the *Informatica Data Replication Command Line Interface for the Server Manager*.

Replication Configuration Command Line Interface

The Replication Configuration Command Line Interface (CLI) is a command line utility that you can use to update source and target metadata for replication configurations. The configurations must have been created in the Data Replication Console and saved as XML files.

Stages of Replication Processing

Replication processing consists of multiple stages. For example, in a topology that runs a Server Manager instance on the source system and on the target system, replication processing occurs in the following stages:

1. The Extractor captures changes from the source database and writes the changes to intermediate files. Then the Extractor sends a message to the Server Manager on the source system that confirms the extraction stage is complete.
2. The Server Manager instance on the source system notifies the Server Manager on the target system that the intermediate files are ready to be transferred.
3. The Server Manager instance on the target system uses HTTP or HTTPS to get the intermediate files and then sends a message to the Server Manager on the source system to confirm that the files were transferred.
4. The Server Manager instance on the source system removes the intermediate files that were transferred to the target.

   Note: For replication jobs with multiple targets, the Server Manager instance waits until the intermediate files have been transferred to all of the targets.
5. The Applier applies the changes from the intermediate files to the target database. When apply processing is complete, the Applier sends a confirmation message to the Server Manager instance on the target. The Server Manager purges the intermediate files for which all changes were processed and retains only the intermediate files that contain uncommitted transactional changes.

Alternative Deployment Topologies

Data Replication supports alternative data replication topologies to meet your replication needs.

Common topologies are:

- **One source to one target**. Replicates data from a single source to a single target.
- **One source to multiple targets**. Replicates data from a single source to multiple targets.
- **Multiple sources to one target**. Replicates data from multiple sources to a single target.
- **Bidirectional replication**. Replicates data bidirectionally for a pair of databases.
- **Cascade replication**. Replicates data across a series of databases.

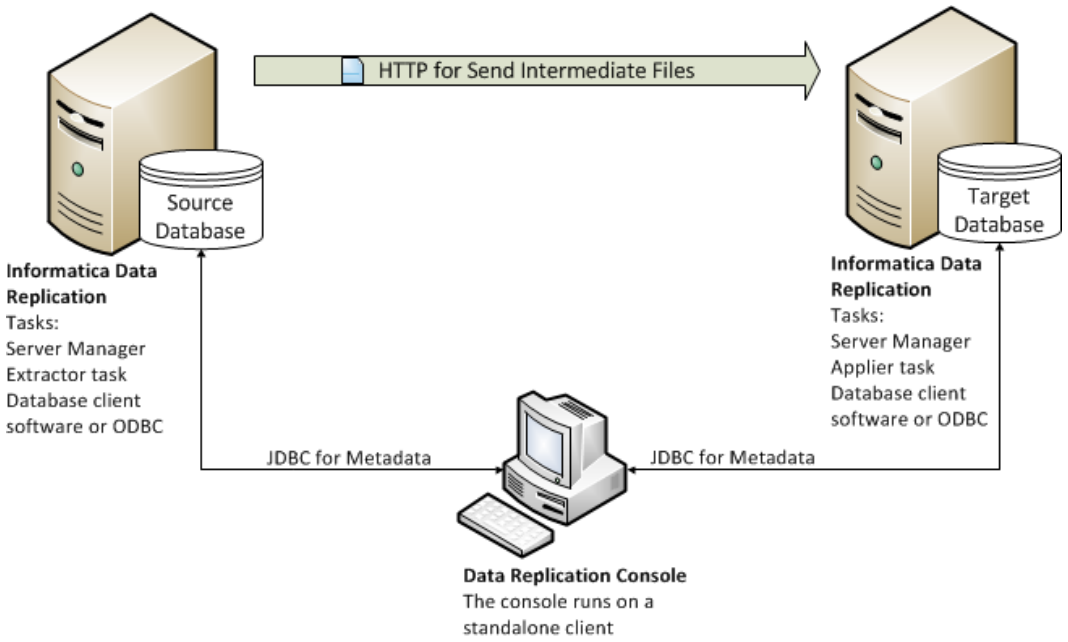
Data Replication is flexible and adapts to your network configuration. You can deploy Data Replication components on a single system for local replication or on different systems for distributed replication across your environment.

Review the sample network topologies to learn where to install Data Replication components under alternative scenarios.

Replication from a Single Source to a Single Target

Typically, you run the Extractor and a Server Manager instance on the system with the source database and run the Applier and a Server Manager instance on the system with the target database. Data Replication transmits the intermediate files between the source and target systems by means of a LAN or WAN. You can run the Data Replication Console as a standalone client or on the source or target system.

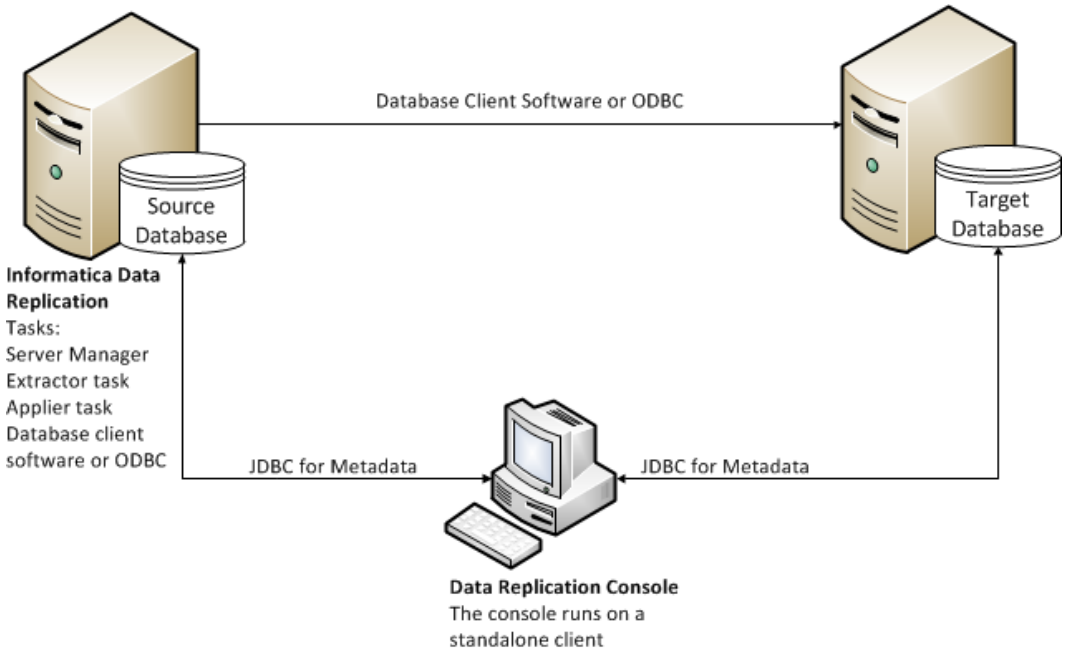
The following image shows the typical deployment topology:



In this topology, you run the Extractor and a Server Manager instance on the source and run the Applier and a Server Manager instance on the target. The Data Replication Console optionally runs on a standalone machine.

However, you do not have to deploy Informatica Data Replication on both the source system and target system.

The following image shows a deployment topology that runs the Applier on the source system:



Related Topics:

- "Creating Replication Configurations" on page 174

Replication from a Single Source to Multiple Targets

You can replicate data from a single source database to multiple target databases on different systems.

Note: This topology is not supported for Amazon Redshift or Apache Kafka targets.

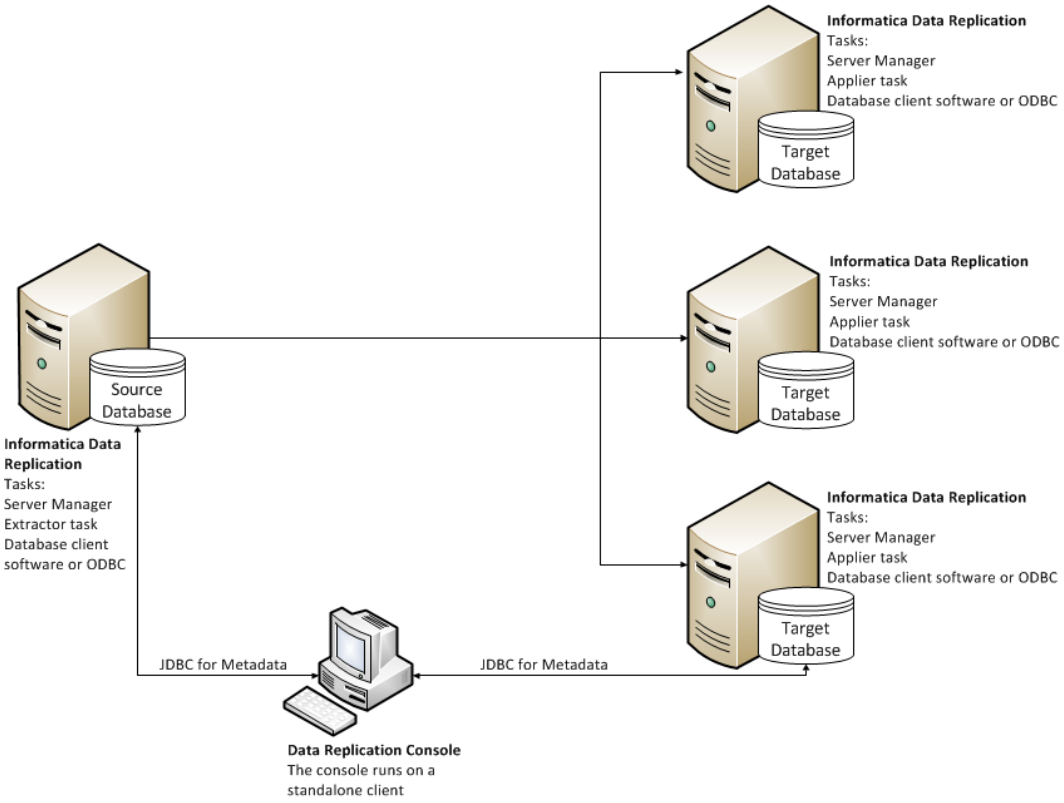
Typically, you install Data Replication on the source system and on each target system. On the source machine, run an Extractor task and a Server Manager instance. On each target machine, run an Applier task and a Server Manager instance. Each target requires a separate Applier task and Server Manager instance.

If the resources on a target machine are constrained, you can run the Applier task and Server Manager instance for that target on another machine that can access the target database by using the native target database connectivity client or and ODBC driver.

One of the target databases is considered to be the *primary target database*. The other *secondary target databases* must be the same target type as the primary target. Also, all of the mapped target tables must have identical schema definitions. However, the target schemas can have different names.

Note: The primary target database is defined on the **Target Database** tab in the Data Replication Console. The secondary target databases are defined on the **Routing** tab in the same replication configuration.

The following image shows a typical topology that includes three targets, each with an Applier task and Server Manager instance, and the Data Replication Console on a standalone machine:



Related Topics:

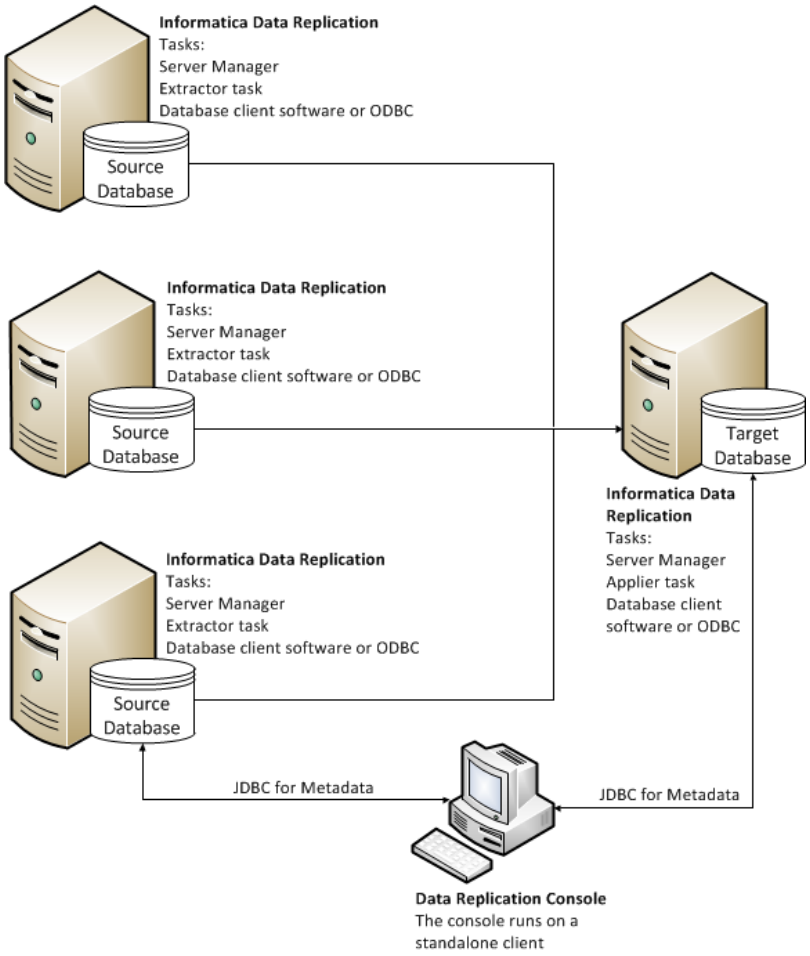
- "Configuring Data Replication from One Source to Multiple Targets" on page 297

Replication from Multiple Sources to a Single Target

You can replicate data from multiple source databases on different systems to a single target. You can use source databases of different types when replicating from multiple sources to a single target.

To replicate data from multiple sources, you typically install Informatica Data Replication on each source system and on the target system. On each source, run an Extractor task and a Server Manager instance. On the target, run an Applier task and a Server Manager instance.

The following image shows a typical deployment topology that includes three sources and the Data Replication Console on a standalone machine:



Alternatively, you can run the Extractor and Server Manager on a separate machine that can access the source database logs. Also, you can run the Applier and Server Manager on any machine that can access the target database by means of the target database connectivity client or ODBC driver. For example, run the Applier on one of the source systems. In this case, you would not install Data Replication on the target system.

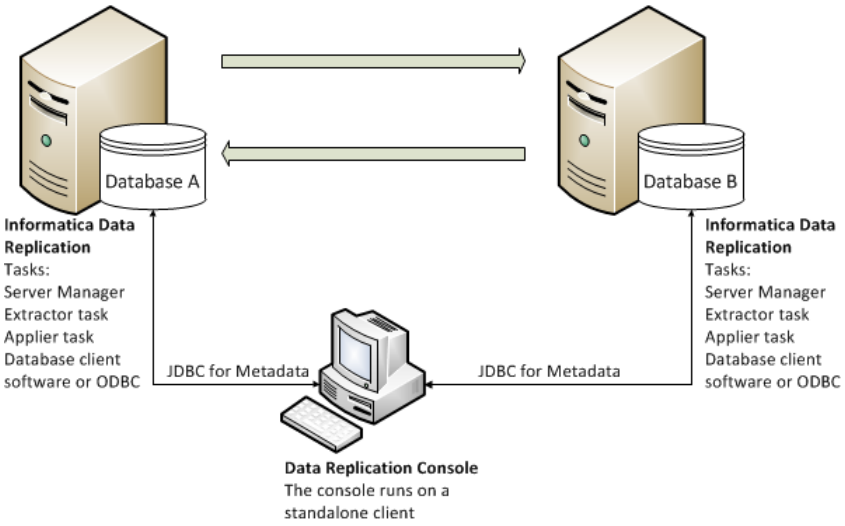
Related Topics:

- "Conflict Resolution for Replicated Data" on page 41
- "Deployment Overview" on page 349

Bidirectional Replication

In a bidirectional replication topology, Data Replication replicates changes between two systems in both directions simultaneously. Users can write changes to both databases.

The following image shows a typical deployment topology for bidirectional replication:



You must install Informatica Data Replication on both systems. Run an Extractor task, Applier task, and Server Manager instance on each system. You can run the Data Replication Console on a standalone system.

Data Replication simultaneously replicates changes that occur on Database A to Database B and changes that occur on Database B to Database A.

Note: Bidirectional replication is not supported for MySQL source databases or Apache Kafka target messaging systems..

Data Replication maintains data integrity by providing loopback avoidance and conflict resolution.

Related Topics:

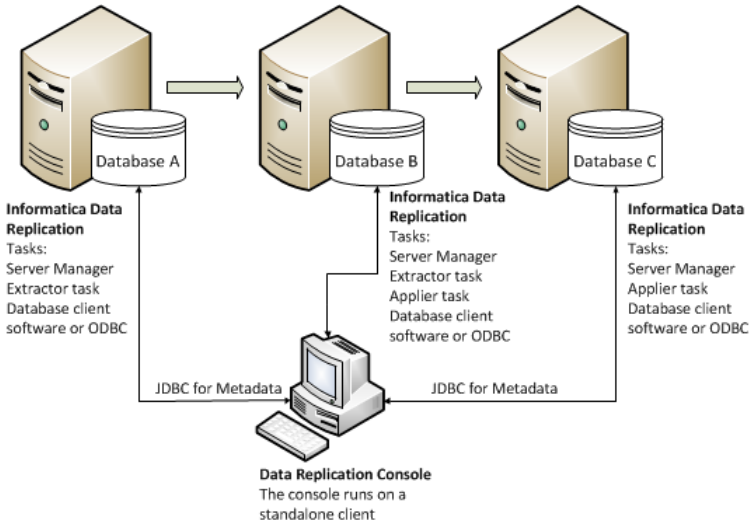
- "Conflict Resolution for Replicated Data" on page 41
- "Loopback Avoidance for Replicated Data" on page 310
- "Configuring Bidirectional Replication" on page 307

Cascade Replication

In a cascade replication topology, Data Replication replicates change data unidirectionally down a series of databases. Users can write changes to any database in the chain.

A database in a middle of the chain can act as both a source and target simultaneously. From any database in the chain, Data Replication replicates forward all changes that originally occurred on that database to the other databases in the chain.

The following image shows a typical cascade replication topology that includes three databases:



In this deployment topology, you must run a Server Manager instance on each system. Run an Extractor task on each system from which you capture database changes and replicate changes forward (Database A and Database B). Run an Applier task on each system to which you apply changes (Database B and Database C). You can run the Data Replication Console on a standalone system.

When the cascade replication runs, Data Replication replicates forward the changes that occurred on Database A to Database B. When replicating changes from Database B to Database C, Data Replication replicates both the changes that originated on Database A and Database B.

Note: Data Replication does not support cascade replication for Apache Kafka targets.

Related Topics:

- "Loopback Avoidance for Replicated Data" on page 310
- "Configuring Cascade Replication" on page 309

Replication Configurations

A replication configuration contains the information that is required to run the InitialSync, Extractor, and Applier tasks to materialize targets or replicate changes to targets.

A configuration includes the following types of information:

- Connection information for the source and target databases
- The source and target tables selected for change data replication or target materialization
- Mappings between source and target tables
- Mappings between source and target columns
- Column-filtering conditions
- Locations of the InitialSync, Extractor, and Applier executables and the Extractor output directory
- Various runtime and advanced configuration settings

You can use the Data Replication Console to create replication configurations. The Console guides you through the replication creation process, lists valid options, and validates some entries.

The Data Replication Console stores configurations as SQLite .db files on the system where the Server Manager Main server runs. Each configuration has an owner that is specified when the configuration is created.

If you use the Replication Configuration Command Line Interface (CLI) to update a configuration, export the configuration as an XML file from the Data Replication Console to save your changes. From the **Server Manager** > **Configs** view, click the **Export** icon button on the Replication Configurations toolbar.

You can run the replication tasks with configurations that are stored in an XML or SQLite .db configuration file.

The config.xsd file is an XML schema file that defines the format of the XML configuration. Do not edit this schema file.

For the SQLite .db configurations, Data Replication stores all revisions of the configuration.

CHAPTER 2

Understanding Data Replication

This chapter includes the following topics:

Overview

Before you create replication configurations, become familiar with key replication concepts on the following topics:

- Start Point values for the Extractor
- How to handle source tables that do not have a primary key definition or a unique index
- Apply processing
- Checkpointing
- Recovery
- Use of Tcl and SQL scripts
- Generated virtual Indexes
- Database character set conversion
- DDL replication considerations and examples

Replicating Source Tables That Do Not Have a Primary Key Definition

Data Replication requires rows to be uniquely identified to accurately replicate Update and Delete operations.

For the source tables that do not have a primary key definition or a unique index, you can either manually create a virtual index for the source table or disable replication of Update and Delete operations. To manually create a virtual index in the Data Replication Console, select a mapped source table on the **Map Tables** tab and click the **Show Indexes** icon button. On the **Indexes** dialog box that appears, select the **Primary Key** check box.

When you save a configuration that extracts Update and Delete operations from a mapped source table that does not have a primary key definition or a unique index, the Data Replication Console prompts you to select one of the following options for replicating change data:

- Create a virtual index that includes all of the mapped table columns. If a mapped column has a datatype that is not supported for additional logging, the Data Replication Console does not include this column in the virtual index.
- View a list of source tables that do not have a primary key definition on the **Map Tables** tab. You can then assign a primary key or unique index definition to these tables in the source database.

  Tip: Later, when you want to view a full list of the source and target tables again, click **Refresh**.
- Disable replication of Update and Delete operations. In this case, Data Replication replicates only Insert operations. Also, for DB2, SQL Server, and Oracle sources, Data Replication replicates DDL changes.

Start Point for the Extractor Task

A Start Point indicates the point from which the Extractor starts reading transaction or event logs when you run the Extractor task for the first time with a particular configuration.

For DB2, Microsoft SQL Server, and Oracle sources, the Extractor reads transaction logs. For MySQL sources, the Extractor uses the mysqlbinlog utility to read change events from the binary log.

How the Start Point value is represented depends on the source type:

- For DB2 and Microsoft SQL Server sources, the Start Point is an LSN value.
- For Oracle sources, the Start Point is an SCN value.
- For MySQL sources, the Start Point is identified by log coordinates, which consist of the binary log file name with a numeric suffix and an offset position in the file.

By default, the Extractor starts reading logs from one of the following points, depending on the source type:

- For DB2 sources, you run the Extractor twice if you do not override the default Start Point value for the Extractor. When you run the Extractor the first time, it determines the current LSN value and writes it to the db2.initial_lsn parameter in the configuration. When you run the Extractor the second time after performing an InitialSync, the Extractor starts reading the transaction logs from the LSN value that is specified in the db2.initial_lsn runtime parameter.

- For Oracle and Microsoft SQL Server sources, the Extractor starts reading the transaction logs from the SCN or LSN that is the least of the following two values:
  - The current Start Point value at the time the Data Replication Console retrieves metadata for the configuration from the source database.
    To determine the current LSN for Microsoft SQL Server sources, Data Replication opens a transaction and then creates and drops an auxiliary table in this transaction. The default name of the auxiliary table is IDR_DUMMY_2. To use another name for the auxiliary table, add the mssql_auxiliary_table_name_2 parameter in the *DataReplication_installation*/uiconf/default.cfg file.

    Note: If the default.cfg file does not exist, use a text editor to create the file in the uiconf directory and add the mssql_auxiliary_table_name_2 parameter in the file. For more information, see "Default.cfg File" on page 391.
  - The lowest Start Point value for any open transactions at the time the Data Replication Console retrieves metadata from the source database. The Data Replication Console retrieves metadata when you select a source database or schema on the **Map Tables** tab.
- For MySQL sources, the first time you start the Extractor after an InitialSync run, the Extractor begins reading events from the binary log at the log position for the initial synchronization event.

You can override the default Start Point value in the configuration to change the default Extractor behavior and process the transaction logs from a particular point. For example, you might need to set the Start Point value under the following circumstances:

- To reprocess the transaction or binary logs that the Extractor processed earlier for debug purposes.
- To skip the transaction or binary logs that do not include changes for the mapped source tables.
- To read the transaction or binary logs from a point that is earlier than the default Start Point. For example, if you created a replication configuration after you synchronized targets using Fast Clone for Oracle sources, you can configure the Extractor to read changes that occurred after you completed target synchronization and before you created the configuration in the Data Replication Console.

The following considerations apply to the Extractor Start Point processing:

- The Extractor skips any transaction logs that have lower SCN or LSN values than the logs that include the specified Start Point. For MySQL sources, the Extractor skips any binary logs older than the log that includes the log coordinates for the Start Point position.
- If the transaction log or binary log that includes the Start Point is not available but logs that include greater SCN, log coordinate, or LSN values are available, the Extractor reports an error and stops processing by default.
  To recover from the error, either make the appropriate log available to the Extractor or specify a greater Start Point value.

  Alternatively, for Microsoft SQL Server and Oracle sources, you can configure the Extractor to skip the missing log by setting the SKIP_CONTINUITY_CHECK command-line parameter to **Y**. The Extractor then starts reading data from the available database log that has the lowest LSN or SCN value. However, Informatica recommends that you specify a greater Start Point value instead to avoid data integrity problems on the target database.
- After you run replication tasks for MySQL sources, Informatica recommends that you do not change the base name of the binary log files or change the name or location of the directory that contains the log files as specified on the **Extract Range** tab. If you must change the base log file name or change the binary log directory name or location, perform the following actions:
  1. Stop all running replication tasks.
  2. To change the base name of the log files, rename the existing log files that were generated with the previous base name.

3. To change the directory name or location, copy the existing binary log files to the new directory. Then update the path on the **Extract Range** tab.
4. Resume replication.

- For DB2 and SQL Server sources, if you set the Start Point value to an LSN that is in the middle of a long-running transaction, the Extractor skips all records in the transaction.
- For accurate replication of data from DB2, SQL Server, and Oracle sources, set the Extractor Start Point to a value greater than or equal to the LSN or SCN corresponding to the point in the logs at which you enabled supplemental logging.
- For Oracle and SQL Server sources, if you set the extract.process_old_logs runtime parameter to 1, the Extractor ignores the Start Point value and processes all available transaction logs.

Related Topics:

- "Configuring the Start Point for the Extractor Task" on page 226

Apply Processing

Apply Modes

The Data Replication Applier provides the following apply modes: SQL Apply, Audit Apply, and Merge Apply. Data Replication preserves the correct order of the changes that are spread across threads by using internal locks based on row identifiers.

SQL Apply

SQL Apply executes SQL statements on the target that are equivalent to the SQL change operations on the source. For every Update, Insert, or Delete operation on the source, the Applier writes a corresponding Update, Insert, or Delete on the target.

Note: You must perform an initial synchronization of targets prior to starting change data replication in SQL Apply mode.

Data Replication does not support SQL Apply mode for Amazon Redshift, Apache Kafka, Flat File, Greenplum, Cloudera, Hortonworks, Netezza, Teradata, and Vertica targets.

Audit Apply

For relational database and data warehouse targets, you can use Audit Apply mode to audit SQL changes made to a source by using an audit log table in the target database.

For Apache Kafka, Cloudera, Flat File, and Hortonworks targets, Audit Apply is the only available apply mode that you can use to replicate data and metadata. The target output files and Kafka messages contain data and metadata similar to an audit log table.

Note: With Audit Apply mode, you do not need to perform an initial synchronization of targets prior to starting change data replication.

Related Topics:

- "Audit Log Tables" on page 34
- "Generating Target Tables and Audit Log Tables Based on Source Table Schema" on page 199

Merge Apply

With Merge Apply mode, each source table has a corresponding target table and audit log table on the target. The audit log tables must be based on the target tables, not on the source tables.

You must perform an initial synchronization of targets prior to starting change data replication in Merge Apply mode.

The Applier accumulates all changes for an apply cycle in intermediate audit log tables, merges the changes into fewer SQL statements, and then applies the changes to the target tables. The audit log tables are truncated at the beginning of each apply cycle.

You can use Merge Apply for data replication to Amazon Redshift, Greenplum, Netezza, Oracle, Teradata, and Vertica targets.

Data Replication does not support Merge Apply mode for Apache Kafka, DB2 for Linux, UNIX, and Windows,Flat File, PostgreSQL, MySQL, and Microsoft SQL Server targets.

To replicate Updates and Deletes, the source tables must have a logical primary key. Data Replication uses this logical primary key to uniquely identify rows on the target. If source table definitions include primary keys or unique indexes, Data Replication uses these keys or indexes as the logical primary key. If a source table does not have a primary key or unique index, you must define a virtual index for the table from the Data Replication Console.

Note: If primary key columns in replicated source tables contain null values, the processing of Updates and Deletes in Merge Apply mode might degrade Data Replication performance.

The SQL merge statements are optimized for the types of SQL changes in the apply batch. For example, if the apply batch contains only Inserts for a table, the merge operation uses a simple INSERT INTO SELECT statement. If the apply batch contains Updates and Deletes for a table, two joins are performed on the target table: one join for the Updates and one join for the Deletes.

Important: Data Replication does not enclose table and column names in double quotation marks in the SQL merge statements. Ensure that the names of target schemas, tables, and columns do not match any reserved words of the target database. Also, for Oracle, Netezza, Greenplum, and Vertica targets, ensure that the names of the target schemas, tables, and columns use the default name case.

For Netezza, Oracle, and Teradata targets, the maximum row size that the Applier can apply in Merge Apply mode is 272 KB.

For Oracle targets, you can use SQL Apply mode or Merge Apply mode. Informatica recommends that you use SQL Apply mode because it typically provides better performance. However, if the replicated tables contain bitmap indexes, Merge Apply mode is faster than SQL Apply mode.

Important: Data Replication does not support secondary unique indexes for targets that use Merge Apply mode. To use Merge Apply mode, you must either drop uniqueness constraints for the secondary indexes on the target database or create initially deferred unique constraints. Use the following SQL statements:

- To drop a uniqueness constraint for the secondary index:

```
ALTER TABLE table_name DROP CONSTRAINT constraint_name;
```

- To create an initially deferred unique constraint:

```
DROP INDEX index_name;
ALTER TABLE table_name ADD CONSTRAINT constraint_name UNIQUE (column_names)
DEFERRABLE INITIALLY DEFERRED;
```

Audit Log Tables

Data Replication uses audit log tables in Audit Apply and Merge Apply modes for targets other than Apache Kafka, Cloudera, Flat File, and Hortonworks. Also, InitialSync uses audit log tables to process SQL expressions in SQL Apply mode when using the Bulk Copy Program (BCP) to load data to Microsoft SQL Server targets. An audit log table records a row for every SQL Update, Insert, or Delete operation on a source table.

Each row in an audit table stores metadata about an SQL operation and the before- and after-images of the data for each column in the mapped table.

Note: For Apache Kafka, Cloudera, Flat File, and Hortonworks targets, which must use Audit Apply mode, Data Replication records data and metadata in Kafka messages or in output files instead of in audit log tables.

The following table describes the columns that an audit log table contains:

Column	Description
OP_XID	The transaction ID.
OP_CODE	The code for the operation type.
OP_TIME	The date and time of the operation.
OP_CMT_SCN	The commit SCN.
OP_CMT_TIME	The commit time of the SQL operation.
OP_NUM_IN_TX	The sequence number of the SQL operation in a transaction.
OP_KEY_LEVEL	The recursive level for row key changes. Data Replication uses this column to process primary key changes in Merge Apply mode.
OP_ROOT_KEY_ROWID	The root ID for recursive key changes. Data Replication uses this column to process primary key changes in Merge Apply mode.
database_user_ID	The user ID of a database user who performed a DML operation, as recorded in the transaction log record for the operation. **Note:** Data Replication includes this column in audit log tables only if you entered the column name in the **User ID** field on the **Runtime Settings** tab > **Calculated Columns** view.
database_user_name	The name of a database user who performed a DML operation, as recorded in the transaction log record for the operation. **Note:** Data Replication includes this column in audit log tables only if you entered the column name in the **User name** field on the **Runtime Settings** tab > **Calculated Columns** view.
<*column_name*>_OLD	The before-image value of the specified mapped source column in rows for Update and Delete operations. For Insert operations that are processed in Merge Apply mode, the Applier copies the value from the <*column_name*>_NEW column for the primary key to the <*column_name*>_OLD column to simplify merge-apply processing. **Note:** In the name <*column_name*>_OLD, the <*column_name*> value must match either the name of a source column if you are using Audit Apply mode or the name of a target column if you are using Merge Apply mode.

Column	Description
<*column_name*>_NEW	The after-image value of the specified mapped source column in rows for Insert and Update operations. In the name <*column_name*>_NEW, the <*column_name*> value must match either the name of a source column if you are using Audit Apply mode or the name of a target column if you are using Merge Apply mode.
OLD_<*column_name*>	The before-image value of an unmapped source column that is used in a SQL expression associated with a virtual column. Data Replication uses the OLD_<*column_name*> column when processing SQL expressions in Merge Apply mode. The names of the columns that contain before- and after-images of Insert, Update, and Delete operations must match the names of the source columns for Audit Apply mode or match the names of the target columns for Merge Apply mode.
NEW_<*column_name*>	The after-image value of an unmapped source column that is used in a SQL expression associated with a virtual column. Data Replication uses the NEW_<*column_name*> column to process SQL expressions in Merge Apply mode. The names of the columns that contain before- and after-images of Insert and Update operations must match the names of the source columns for Audit Apply mode or match the names of the target columns for Merge Apply mode.

On the **Runtime Settings** tab > **Calculated Columns** view, you can change the names of these metadata columns and the suffixes that Data Replication appends to the names of the columns for before images and after images. The default suffixes are _OLD and _NEW.

Note: On the **Calculated Columns** view, you can override the default suffixes in the **Before image column name ending** and **After image column name ending** fields. At least one of these fields must specify a suffix. If both fields are empty, the Applier cannot identify the before-image and after-image columns in the audit log table.

Data Replication generates the prefixes for OLD_<*column_name*> and NEW_<*column_name*> columns that are associated with SQL expressions based on the suffixes that are specified for the before-image and after-image columns. To create a prefix from a suffix, Data Replication removes the leading underscore character, if any, and appends the underscore character to the end of the suffix value. By default, Data Replication uses the suffixes _OLD and _NEW and the prefixes OLD_ and NEW_.

Important: For Audit Apply mode, if you add a column to the audit log table using a source column name and the audit log table already contains generated columns that match against that source column name, the Data Replication Console unmaps the audit log table in the configuration. For example, you add column PK to the audit log table and the source table contains a column of the same name. Also, the audit log table already contains the columns PK_NEW and PK_OLD, which were generated for the source column. In this case, the column mappings become invalid and the audit log table is unmapped.

From the Data Replication Console, you can generate audit log tables based on source or target schemas:

- For Audit Apply mode, use the source schema to generate audit log tables.
  The name of an audit log table can differ from the source table name. However, in this case, you must map the source table with the target audit log table manually.

  Tip: If you plan to use virtual columns with SQL expressions in Audit Apply mode, Informatica recommends that you manually add an index on the combination of the OP_CODE and OP_CMT_SCN columns in each audit log table for faster processing of Update operations that the Applier runs to apply SQL expressions to the loaded table rows.
- If you use InitialSync with the Bulk Copy Program (BCP) to load data to Microsoft SQL Server targets and require audit log tables for InitialSync to process SQL expressions defined for source tables in SQL Apply

mode, use the target schema to generate audit log tables. The names of the audit log tables and columns must be based on the names of the target tables and columns.

- For Merge Apply mode, use the target schema to generate audit log tables. The names of the audit log tables and columns must be based on the names of the target tables and columns.
  The following additional considerations apply to audit log tables for Merge Apply:
  - If you use SQL expressions, first map virtual columns to target columns and then generate the audit log tables based on the target schema.
  - An audit log table name must match the corresponding target table name and end with the log suffix that is specified in the **Log table suffix for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view. The default audit log table suffix is _LOG.
  - If you generated the target table from the Data Replication Console, you can generate an audit log table by using the source or target schema. However, if you created the target table manually outside of the Data Replication Console, you must generate the audit log table for the target table based on the target table schema.
  - If you want to use a schema other than the target schema to create the audit log tables, specify this schema in the **Log table schema for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view.

Related Topics:

- "Audit Apply" on page 32
- "Generating Target Tables and Audit Log Tables Based on Source Table Schema" on page 199

Kafka Message Structure

The Kafka messages to which Data Replication writes change data and metadata use an Apache Avro schema, which is similar to the audit log table schema, to define the structure of Kafka messages.

Data Replication creates one Kafka message for each source database operation and then publishes each message to a configured Kafka topic. For Update operations, Kafka messages contain both before and after images of the source columns. The before-image field names consistent of the source column names plus a suffix and optional prefix, whereas the after-image field names match the source column names.

Data Replication produces an Avro schema file for each replicated source table. Each schema file lists a sequence of Avro fields for the corresponding Kafka message. The following table describes the Avro fields in an Avro schema file:

Avro Field Name	Description
OP_TIME	The date and time of an operation
OP_CODE	The code for the operation type
OP_CMT_SCN	The commit point represented by a source SCN, log coordinates, or LSN
OP_CMT_TIME	The commit time of the operation
OP_XID	The transaction ID
OP_NUM_IN_TX	The sequence number of an operation in a transaction

Avro Field Name	Description
*<prefix><source_column_name>*_OLD	The before-image value of the source column. The default suffix in this column name is _OLD. For this field to be generated, the apply.avro.avro_include_before_image setting must be 1. You can change the prefix and suffix values by setting the apply.avro.avro_before_image_prefix and apply.avro.avro_before_image_suffix runtime parameters.
*<prefix><source_column_name>*_PRESENT	The indicator that a null value in an Avro field corresponds to a null value in the source column. The default suffix in this column name is _PRESENT. For this field to be generated, the apply.avro.avro_include_is_present setting must be 1. You can change the prefix and suffix values by setting the apply.avro.avro_is_present_prefix and apply.avro.avro_is_present_suffix runtime parameters.

Data Replication supports only the following primitive Avro datatypes in Kafka messages:

- NULL
- BOOLEAN
- INT
- LONG
- BYTES
- STRING

Data Replication converts all source data to characters in Kafka messages. Data Replication does not support the FLOAT or DOUBLE primitive Avro datatype because converting data in a source FLOAT or DOUBLE datatype to characters and then to the Avro FLOAT or DOUBLE datatype could decrease data accuracy and precision.

Related Topics:

- "Generating Avro Schemas for Apache Kafka Consumers" on page 203

Applier Handling of the Sync Point Value

The Applier uses the Sync Point value in the configuration to filter DML operations.

When you run the Applier, it skips all DML changes that have an SCN value, LSN value, or log coordinates less than or equal to the Sync Point value in the configuration. This behavior ensures data consistency.

If you run batched replication, the Applier issues an error if the configuration includes one or more tables for which the Sync Point value is -1. If you run continuous replication, the Applier skips all of the changes for the tables for which the Sync Point value is -1.

Related Topics:

- "DBSYNC_SYNC_LSN Table" on page 64
- "Sync Point Value" on page 272

Applier Processing of Intermediate Files

The Applier run might include multiple apply cycles. During an apply cycle, the Applier processes intermediate files and commits the changes to the target at the end of the cycle.

During each apply cycle, the Applier processes one or more intermediate files depending on the apply.process_intermediate_size_per_job parameter value. This parameter determines the number of intermediate files that the Applier processes during an apply cycle. If this parameter is set to 0, the Applier processes all available intermediate files. If this parameter is set to a value greater than 0, the parameter specifies the maximum total size of all intermediate files, in megabytes, that the Applier processes during a single apply cycle. Data Replication always processes entire intermediate files. Data Replication never splits an intermediate file to avoid exceeding the maximum total size that is specified in this parameter. You control the maximum size of a single intermediate file by setting the **Maximum size of each intermediate file** option on the **Runtime Settings** tab > **General** view.

An intermediate file is composed of a data file (.dat) and a transaction file (.trn). The .trn files contain transaction metadata. The .dat files contain the transaction data changes and can be very large.

When the Applier processes the intermediate files during an apply cycle, it looks for a commit in the .trn files. After the Applier encounters a commit in a .trn file, the Applier starts reading the corresponding .dat files. If the Applier does not encounter a commit in the .trn files during the current apply cycle, the Applier queues the corresponding .dat files. Then, whenever the Applier encounters a commit during a subsequent apply cycle, it processes all of the queued .dat files.

When the Applier processes a .dat file, it applies all committed transactions to the target. The Applier accumulates changes that belong to open transactions in memory buffers. Change data for each long-running transaction is stored in a separate buffer. After the Applier encounters a commit for a long-running transaction during a subsequent apply cycle, the Applier applies the data from the corresponding buffer to the target database and then clears the buffer.

For target data warehouse appliances that restrict the number of load connections, the Applier concurrently loads data to a batch of target tables. The number of tables in a batch cannot exceed the number of available Applier threads. For the remaining tables, the Applier accumulates change data from each source table in a separate memory buffer. After the Applier loads the data for the batch of tables to the target, it reads the data for the next batch of tables from the corresponding buffers. After the Applier loads data to all of the target tables, it commits the changes and finalizes the apply cycle.

The apply.buffer_size_for_split_records runtime parameter specifies the maximum size of the buffers that accumulate data for long-running transactions and target tables. If the amount of data in the buffer exceeds the specified limit, the Applier flushes the data from the buffer to a temporary spill file in the *DataReplication_installation*/output/*configuration_name*/tmp directory. The Applier then writes subsequent changes to the spill file instead of to the buffer. The spill file names have the following formats:

- For long-running transactions: *configuration_name_transaction_xid*.spill
- For target tables: *configuration_name_table_id*.spill

Because the spill files can be large, the Applier does not flush existing spill files to disk when taking checkpoints that are used to resume processing after an outage. Consequently, when the Applier restarts, it deletes existing spill files that might be incomplete and re-reads the intermediate files to process all of the records in a long-running transaction.

Related Topics:

- "Editing an Index for a Source Table" on page 220

Applier Processing of Updates in Merge Apply Mode

In a replication configuration, you can configure the Applier to apply Updates as a pair of Delete and Insert operations in Merge Apply mode. This Applier optimization avoids the overhead of building complex SQL statements for updating primary key columns.

If the primary key columns that the Applier uses to identify rows in a target table are frequently updated on the source, enable this optimization for the table to improve Merge Apply performance. The Applier optimizes processing of Updates if the mapped table has a virtual index that meets the following requirements:

- The virtual index includes all of the table columns.
- In the **Indexes** dialog box of the Data Replication Console, the **Unique** option is selected for the virtual index.
- In the **Indexes** dialog box of the Data Replication Console, the **Supplemental logging** option is selected for the virtual index. This option enables additional database logging for all of the table columns. The Applier requires the before- and after-image values for all of the table columns to build the Delete and Insert statements.

If a virtual index that meets these requirements does not exist, create one. The Applier requires this virtual index to identify the target rows to which to apply the pair of Delete and Insert operations for an Update.

If a mapped table has other virtual indexes for which both the **Unique** and **Supplemental logging** options are selected, clear one of these options for each of the virtual indexes. By clearing an option, you increase the priority of the unique index that includes all of the table columns and decrease the priority of the other virtual indexes. The Applier then uses the unique index with the highest priority to identify the target rows when applying a pair of Delete and Insert operations for optimized Update processing.

Note: You cannot use this optimization if the source table includes columns for which the before- and after-image values are not available in the database log.

Applier Processing of Updates for Duplicate Records

If mapped source and target tables contain duplicate records, the Applier processing of Updates for these duplicate records depends on the apply mode.

- In Audit Apply mode, the Applier processes Updates for duplicate records correctly.
- In Merge Apply mode, the Applier ends with an error when processing Updates for duplicate records.
- In SQL Apply mode, the Applier processing of Updates for duplicate records depends on whether the updated columns are part of a virtual index:
  - If the updated column in duplicate records is part of a virtual index, the Applier updates all of the target duplicate records.
  - If the updated column in duplicate records is not part of a virtual index, the Applier always updates only one of the target duplicate records. This behavior results in data inconsistencies.

  Tip: To avoid data inconsistencies between the source and target tables that might contain duplicate records, add a virtual index for the source columns that might be updated on the source.

Commit Processing and Target Constraints with Applier Multi-Threaded Processing

The Data Replication Applier can use multi-threaded processing to apply changes to the target. You can control whether the Applier uses a single transaction across all threads or a transaction for each main thread or each subtask on a main thread.

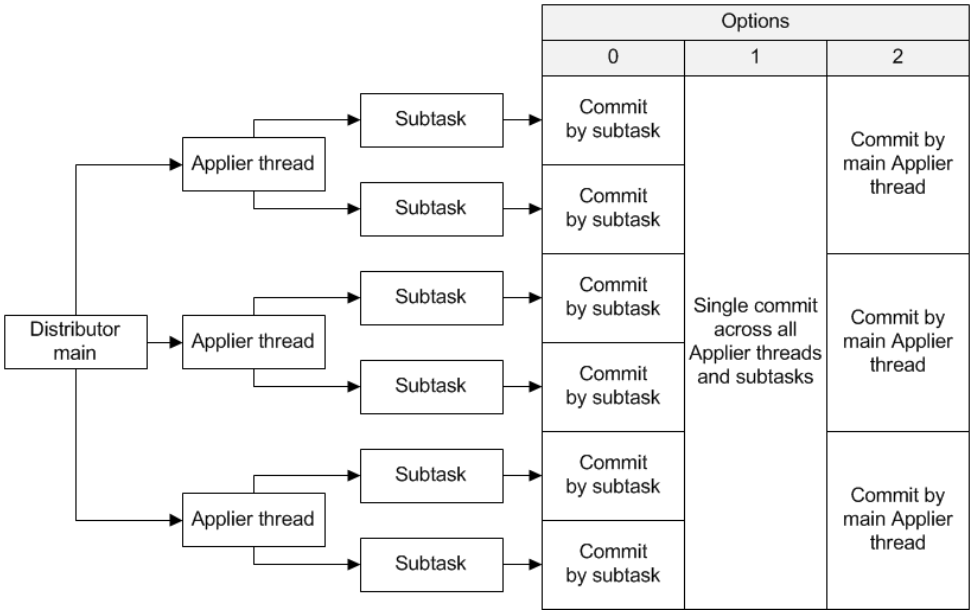
When the Applier uses multiple threads, target table constraints might cause the Applier to fail if some data must be committed before other data. For example, if you have two target tables with a primary key–foreign key relationship and add a row to the first table that references a nonexistent key in the second table, the Applier fails.

To configure commit processing across threads, use the following options:

- To enable Applier multi-threaded processing, on the **Runtime Settings** tab > **General** view, enter a number greater than 1 in the **Applier threads** field. The Applier then commits data on each Applier thread.
- For additional parallelism, on the **Runtime Settings** tab > **Advanced Settings** view, you can set the apply.subtasks_enabled parameter to 1 to enable subtask threads on each main Applier thread. The Applier then commits data in a transaction on each subtask. If you enable subtasks, the Applier distributes changes across threads by object rather than by row.
- For Oracle targets, on the **Runtime Settings** tab > **Advanced Settings** view, you can use the apply.global_transaction_support parameter to control whether the Applier uses a global transaction. A global transaction contains all of the changes that are distributed across multiple Applier threads. Options are:
  - **0**. Disable global transactions and commit data on each subtask thread or on each Applier thread if you do not use subtasks.
  - **1**. Enable global transactions and perform a single commit for all Applier threads or subtask threads at the end of the apply cycle. This option might degrade performance.
  - **2**. Enable global transactions and perform a commit on each Applier thread.

Note: For Apache Kafka targets, do not set the **Applier threads** field to a number greater than 1 or specify the apply.subtasks_enabled or apply.global_transaction_support runtime parameters. The Kafka Apply process and Queue Adapter maintain the commit order when sending change data to the Kafka target message queue.

The following image summarizes the effects of the apply.global_transaction_support parameter options:



To handle target table constraints, use one of the following strategies:

- For Oracle targets, define the table constraints as deferrable and set the Data Replication apply.global_transaction_support parameter to 1. This configuration forces the Applier to check the target table constraints at the end of the global transaction.
- Disable constraints on the target tables.
- Configure the Applier to use one thread without subtasks.

Note: If you need to change Applier processing settings, such as the number of Applier threads and subtask threads, or change an algorithm for distributing change data across threads, stop the Applier task and wait for processing to complete normally. Then edit and save the configuration and restart the Applier task. If you force the Applier task to end, the Applier will process the applied changes again after you restart it.

Conflict Resolution for Replicated Data

Data Replication provides a conflict resolution mechanism to handle conflicts that might occur when you replicate change data to a database on which other change activity occurs.

If you use the bidirectional replication topology or a replication topology with multiple sources and a single target, you must configure conflict resolution for accurate replication of data. Data Replication supports conflict resolution only in SQL Apply mode.

In the Data Replication Console, define conflict resolution rules for target tables. The Applier uses these rules to resolve mismatches between source and target values.

Data Replication supports conflict resolution for the following target types:

- DB2 for Linux, UNIX, and Windows
- Microsoft SQL Server
- Oracle

The following table describes the types of conflicts that Data Replication handles:

Conflict Type	Description
Delete	Occurs when a row that is referenced in an SQL Delete operation does not exist in the target table.
Uniqueness	Occurs when the Applier applies change data that violates a uniqueness integrity constraint such as primary key or unique index.
Update	Occurs when the row that is referenced in an SQL Update operation has different values on the source and target. When you configure a replication job, you must select the columns for which Data Replication compares source and target values. For these columns, the Data Replication Console enables supplemental logging so that the Applier can get the before-image and after-image values to compare the source and target rows.

Depending on the conflict type, you can select different conflict resolution strategies.

The following table describes the conflict resolution strategies that are available by conflict type:

Conflict Resolution Strategy	Conflict Types	Description
Custom	All	Calls a stored procedure that you previously defined to handle conflicts. You can define a stored procedure by using commands that the SQL Script Engine provides. When an SQL operation on the target table causes conflict, the Applier calls the stored procedure and passes operation metadata and the before and after values for all of the columns for which supplemental logging is enabled as procedure parameters. These parameters are: - **OP_XID.** Transaction ID. - **OP_CODE.** A code for the operation type, which can be D for deletes, I for inserts, and U for updates. - **OP_TIME.** The date and time of the operation. - **OP_CMT_SCN.** The SCN value for the operation commit. - **OP_CMT_TIME.** The commit time for the operation. - **OP_NUM_IN_TX.** The sequence number of the operation in the transaction. - ***column_name*_OLD, *column_name*_NEW.** The before and after values for the column <*column_name*>. These parameters repeat for each column for which you enabled additional logging. For more information about the SQL Script Engine commands, see the *Informatica Data Replication Scripting Guide*.
Discard	All	Discards the source values and retains the existing values in the target table.

Conflict Resolution Strategy	Conflict Types	Description
Maximum, Minimum	Update	For these resolution strategies, you must select a resolution column. For conflicting rows, Data Replication compares source and target values in the resolution column. - For the Maximum strategy, if the target value is greater than the source value, Data Replication uses the Discard strategy. Otherwise, Data Replication uses the Overwrite strategy. - For the Minimum strategy, if the target value is less than the source value, Data Replication uses the Discard strategy. Otherwise, Data Replication uses the Overwrite strategy.
Overwrite	- Update - Uniqueness	Overwrites records in the target table with the values from the source table.

Notes:

- If you configure an Update conflict resolution rule and independent Update operations occur on a virtual index column on the source and target at the same time, Data Replication detects the conflict, even if the virtual index column is not specified in the conflict resolution rule.

  For example, if you have a table with a primary key definition that Data Replication uses as a virtual index, Data Replication detects a conflict when an Update is written to the primary key column on the source and target simultaneously.
- When an Update conflict occurs for a virtual index column, the Applier can use only the Custom resolution strategy to resolve the conflict. The Applier skips other resolution strategies because it cannot identify a row on the target for which to resolve the conflict.

  For example, if you have a table without a primary key definition and use a virtual index that includes all of the table columns, you can use only the Custom resolution strategy for Update conflicts.
- If a database does not support additional logging for a column datatype, Data Replication cannot detect Update conflicts for the column. Data Replication also cannot use the before and after values as custom procedure parameters for the column.
- Data Replication does not support conflict resolution for LOB columns. Do not use LOB columns to detect Update conflicts. Also, do not use LOBs as resolution columns when using the Minimum or Maximum resolution strategy.
- The Applier does not lock a target row to apply a conflict resolution rule. You might get unexpected data on the target if changes for the row in conflict occur during the interval between Applier detection of the conflict and application of the conflict resolution rule.

Related Topics:

- "Configuring Conflict Resolution" on page 233
- "Bidirectional Replication" on page 26
- "Loopback Avoidance for Replicated Data" on page 310
- "Replication from Multiple Sources to a Single Target" on page 25
- "Deployment Overview" on page 349
- "Configuring Bidirectional Replication" on page 307

Post-Apply Processing

If you use Merge Apply or Audit Apply, you can configure the Applier to run a set of SQL statements or stored procedure calls on the following targets at the end of each apply cycle:

- Amazon Redshift
- Greenplum
- Netezza
- Oracle
- Teradata
- Vertica

Applier thread 1 runs the set of SQL statements on the target database at the end of each apply cycle after all Applier threads complete applying change data to the target. If the post-apply SQL statements complete successfully, the main Applier thread performs a commit on each Applier thread. If an error occurs, the main Applier thread rolls back the change data that was applied during the current apply cycle.

You can define post-apply SQL statements or stored procedure calls for the targets in the *DataReplication_installation*/uiconf/afterapply.xsl file. In the target database section of this file, create the [AFTER_APPLY] section and define an XSL script that transforms the AFTER_APPLY XML data structure into a set of SQL statements. Also, ensure that the apply.post_apply_script_enabled advanced runtime parameter is set to 1.

The Applier uses Apache XALAN XSL processor V 1.10 to process XSL scripts in the afterapply.xsl file and generate post-apply SQL statements for the target database. The XSL scripts transform the in-memory AFTER_APPLY XML data structure that Data Replication produces at the end of each apply cycle for the targets if the replication configuration includes Merge Apply or Audit Apply mappings.

Examples

The following XSL script runs the test_proc stored procedure on the tgt_db1 Teradata target database and passes the apply cycle ID as an argument to this procedure:

```
[TERADATA]
[AFTER_APPLY]
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="text"/>
<xsl:template match="/">
call tgt_db1.test_proc(<xsl:value-of select="//Cycle/@cycle_id"/>)
</xsl:template>
</xsl:stylesheet>
[END_OF_XSL]
[END_OF_DEST]
```

To run multiple post-apply SQL statements on an Amazon Redshift, Greenplum, Netezza, Teradata, or Vertica target, use a semicolon (;) to separate the SQL statements.

Important: For Teradata targets, you can run only one stored procedure.

The following XSL script runs three INSERT statements:

```
[TERADATA]
[AFTER_APPLY]
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="text"/>
<xsl:template match="/">
INSERT INTO tgt_db1.stat1(total_inserts) VALUES (<xsl:value-of select="sum(//Mapping/
@inserts)"/>);
INSERT INTO tgt_db1.stat2(total_updates) VALUES (<xsl:value-of select="sum(//Mapping/
@updates)"/>);
INSERT INTO tgt_db1.stat3(total_deletes) VALUES (<xsl:value-of select="sum(//Mapping/
```

```
@deletes)"/>);
</xsl:template>
</xsl:stylesheet>
[END_OF_XSL]
[END_OF_DEST]
```

To run multiple post-apply SQL statements on an Oracle target, use an anonymous block. The following XSL script runs two INSERT statements:

```
[ORACLE]
[AFTER_APPLY]
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="text"/>
<xsl:template match="/">
BEGIN
  INSERT INTO tgt_db1.STAT1 VALUES (sysdate,<xsl:value-of select="count(//Mapping)"/>);
  INSERT INTO tgt_db1.STAT2 VALUES (<xsl:value-of select="sum(//Mapping/@inserts)"/>,
                                    <xsl:value-of select="sum(//Mapping/@updates)"/>,
                                    <xsl:value-of select="sum(//Mapping/@deletes)"/>);
END;
</xsl:template>
</xsl:stylesheet>
[END_OF_XSL]
[END_OF_DEST]
```

The first INSERT statement inserts the date and time of an apply cycle and the number of tables that are processed during the apply cycle into the STAT1 table. The second INSERT statement inserts the total number of Inserts, Updates, and Deletes that are processed during an apply cycle into the STAT2 table.

AFTER_APPLY XML Data Structure

The Applier stores the results of the current apply cycle in the AFTER_APPLY XML data structure in memory. Data Replication can use this information to generate post-apply SQL statements for the target database.

The following example XML defines the in-memory AFTER_APPLY XML data structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<AFTER_APPLY>
    <ConfigFile name="config_post_apply.xml"/>
    <Cycle cycle_id="15"/>
    <Mapping src_schema="src_db1" src_table="TAB2"  dest_schema="tgt_db1"
dest_table="TAB2"  type="MergeApply"  inserts="10"  updates="0"  deletes="0" />
    <Mapping src_schema="src_db1" src_table="TAB1"  dest_schema="tgt_db1"
dest_table="TAB1"  type="MergeApply"  inserts="9"  updates="0"  deletes="0" />
</AFTER_APPLY>
```

The following table describes the attributes of the in-memory AFTER_APPLY XML data structure, which you can use in XSL transformation scripts:

Element	Attribute	Description
ConfigFile	name	The name of the XML configuration file.
Cycle	cycle_id	An incremental numeric identifier for the apply cycle.
Mapping	src_schema	The source schema name.
Mapping	src_table	The source table name.
Mapping	dest_schema	The target schema name.
Mapping	dest_table	The target table name.

Element	Attribute	Description
Mapping	type	The type of mapping between the source and target tables that are specified in the src_table and dest_table attributes. This attribute can have one of the following values: - **Normal** for SQL Apply - **AuditLog** for Audit Apply - **MergeApply** for Merge Apply
Mapping	inserts	The number of Insert operations on the source table that the Applier processed during the current apply cycle.
Mapping	updates	The number of Update operations on the source table that the Applier processed during the current apply cycle.
Mapping	deletes	The number of Delete operations on the source table that the Applier processed during the current apply cycle.

Data Visibility Across Multiple Applier Threads

Applier thread 1 runs post-apply SQL statements or stored procedures before the main Applier thread commits the data across all of the Applier threads. Consequently, Applier thread 1 can process only the data that was committed during previous apply cycles and the change data that Applier thread 1 applied to the target during the current apply cycle. Applier thread 1 cannot process change data that other Applier threads applied to the target during the current apply cycle.

Some target databases provide mechanisms for reading uncommitted data that was loaded by multiple threads.

For Teradata targets, you can use the LOCK TABLE FOR ACCESS clause to ensure that the post-apply SQL statements or stored procedures process change data that other Applier threads applied to the target during the current apply cycle.

Note: The LOCK TABLE clause is a part of the DML SQL command. It is not a separate statement.

The following SQL statement inserts the total number of records in the tgt_db1.tgt_tab target table and the current timestamp into the tgt_db1.stat table:

```
LOCK TABLE tgt_db1.tgt_tab FOR ACCESS
INSERT INTO tgt_db1.stat(COUNTER,LOAD_TIME)
SELECT count(*), current_timestamp(0) from tgt_db1.tgt_tab;
```

The total number of records includes the change data that was loaded to this table by all of the Applier threads during the current apply cycle even though this data is not committed yet.

Checkpointing

To provide for flexible recovery from both planned and unplanned exceptions that disrupt replication, Data Replication records checkpoint information separately for the Extractor, Server Manager, and Applier. By recording checkpoint information for each of these components, Data Replication can prevent data loss and ensure data consistency across all of the replication stages.

Data Replication stores checkpoint information in SQLite databases. Data Replication creates a separate SQLite database for the Extractor, Server Manager, and Applier task, where the task runs. These SQLite

databases are in addition to the SQLite databases that Data Replication uses to store configuration file and internal information for the Extractor, Applier, and InitialSync processing.

Note: SQLite is installed as part of the Data Replication installation. Data Replication creates and maintains all of its SQLite databases.

The following events trigger Extractor checkpoints:

- The Extractor reaches the end of a log file, which can be an Oracle archive or online redo log, a MySQL binary log, or a SQL Server backup file.
- The Extractor reads all available records from the source database logs.
- In Continuous mode, the Extractor ends a microcycle of the duration that is specified in the **Continuous Replication Latency** option.
- For Microsoft SQL Server sources, the Extractor reads a chunk of a log file. The chunk size is specified by the extract.mssql.checkpoint_size parameter.
- For DB2 for Linux, UNIX, and Windows sources, the Extractor reads a chunk of a log file. The chunk size is specified by the extract.db2.checkpoint_size parameter.
- For Oracle sources, the Extractor tries to read an online redo log that the Oracle database overwrote with new data.
- For Oracle source instances in a RAC environment, an Oracle instance stops and the Extractor processes all of the redo logs in the thread.

The following events trigger Applier checkpoints:

- The Applier commits data to the target.
- The Applier uses subtask threads to apply changes, and a thread that applies a primary key update writes a commit to the target.

For Apache Kafka targets, the Applier saves the sequence of the last change operation successfully sent to the target as a "checkpoint" to a checkpoint file, provided that you use the default guaranteed delivery mode. If you do not use guaranteed delivery, the Applier writes a checkpoint after each Commit operation. The checkpoint file must exist on the system where the Applier, the Queue Adapter, and a Server Manager instance (Main server or subserver) run. You can change the checkpoint file name or directory by editing the apply.kafka.kafka_checkpoint_file_name and apply.kafka.kafka_checkpoint_file_directory runtime parameters. By default, the checkpoint file name matches the configuration name.

Recovery

Exceptions or failures can occur at any stage in replication processing.

For example, the Applier might unexpectedly disconnect from the target database while applying changes, or the Server Manager might fail to transmit intermediate files over the network.

To help you recover from these failures, Data Replication stores information that the Applier can use to resume change processing without data loss or duplication in the following ways:

- For relational target types, the Applier generates a recovery table on the target. A recovery table stores information that prevents the Applier from replicating previously applied data again when the Applier task restarts. Data Replication supports recovery tables for the following target types: Amazon Redshift, DB2, Greenplum, Microsoft SQL Server, MySQL, Netezza, Oracle, PostgreSQL, Teradata, and Vertica.

- For Apache Kafka targets, the Applier uses a checkpoint file instead of a recovery table to recover from the last checkpoint if an outage occurs. A checkpoint is recorded in the checkpoint file after each message is successfully sent to the target or after each Commit operation, depending on the delivery mode. By default, the Applier uses guaranteed delivery and writes a checkpoint after each message is successfully acknowledged as received by Kafka. This mode avoids message loss or duplication but slows apply processing. If guaranteed delivery is not used, a checkpoint is written after each Commit operation. In this case, duplicate or missing messages might occur on the target.

Note: For Cloudera, Flat File, and Hortonworks targets, Data Replication does not provide a recovery mechanism.

Recovery Tables

The Applier generates recovery tables on targets to store internal service information that prevents the Applier from replicating previously applied data again when the Applier task restarts after a failure.

The Applier uses recovery tables for all target types except Apache Kafka, Cloudera, Flat File, and Hortonworks . For Kafka targets, Data Replicaton uses a checkpoint file instead of a recovery table.

The Applier writes information to recovery tables every apply cycle. When an apply cycle starts, if the previous apply cycle completed successfully, the Applier purges the data in the recovery table. If the previous apply cycle ended abnormally, the Applier uses the data in the recovery table to recover from the failure.

The Applier uses data in the recovery table under the following circumstances:

- The Applier task ended abnormally during the last apply cycle, after a commit occurred on the database and before the Applier wrote checkpoint information to the SQLite checkpoint database.
- The Applier task ended abnormally during the last apply cycle, when processing change data between commit records on subtask threads.

During the recovery cycle, the Applier processes only the intermediate files that were not successfully processed during the previous apply cycle. If the recovery cycle completes successfully, the Applier continues to process the intermediate files.

Typically, the Applier generates recovery tables when it runs for the first time. To generate the recovery table on the target, the Applier must run under a user who has the privileges required to create a table on the target. When you save a configuration file for the first time or deploy a configuration to another Server Manager Main server, Data Replication displays a dialog box in which you can edit the recovery table name and confirm the generation of the recovery table. The default name of the recovery table is IDR_RECOVERY. For a replication job with multiple targets, you can override the recovery table name to specify a different name for each target.

For Netezza targets, Data Replication creates multiple recovery tables on each database. The number of recovery tables for a Netezza database must match the number of Applier threads. Data Replication appends the suffix *_N* to recovery table names on Netezza targets, where *N* is the sequence number of the Applier thread that uses this recovery table.

If an error occurs when the Applier is generating the recovery tables, you can manually define and execute CREATE TABLE statements to create recovery tables. For more information, see Appendix F, “DDL Statements for Manually Creating Recovery Tables” on page 455.

With both recovery tables and checkpointing in place, the Extractor, Server Manager, and Applier can resume processing correctly after different types of failures and planned outages, such as database failures, network failures, system failures, Data Replication upgrades, and configuration information upgrades.

Important: If you use Data Replication 9.6.4 or earlier, the length of the column for configuration names (config) in recovery tables is limited to 64 characters. However, the maximum allowed length of a configuration name is 100 characters. For correct recovery processing, ensure that the first 64 characters in

the config column values are unique. If you have recovery tables to which this limitation applies, you can increase the length of the config column in one of the following ways:

- If the target type allows columns to be resized, resize the config column in the recovery table to 256 bytes.
- Re-create the recovery table in Data Replication 9.7.0 or later by performing the following steps:
  1. Ensure that the Applier is not running and that the recovery table does not contain records with non-zero values in the scn or scn_low columns.
  2. Drop the recovery table.
  3. Restart the Applier.

     The Applier re-creates the recovery table on the target with a config column that uses the VARCHAR(256) datatype.

Tcl and SQL Scripts for Advanced InitialSync and Applier Processing of Data

The Data Replication InitialSync and Applier components use the Tcl Script Engine and SQL Script Engine in the Data Replication scripting engine library to process Tcl scripts and SQL expressions, respectively.

To use the Tcl Script Engine effectively, you must be familiar with the basics of the Tcl scripting language. To use the SQL Script Engine effectively, you must be familiar with SQL scripting for the target database.

You can create Tcl scripts or SQL expressions for the following data replication purposes:

- Customize or convert target column datatypes.
- Perform advanced filtering of source data to apply to the target system.
- Perform mathematical operations on target data.
- Generate customized key values.
- Split a source column into multiple target columns.
- Perform aggregation functions on target data.
- Perform advanced logging during a data replication job.
- Execute SQL statements on the target database during a data replication job.
- Use a variety of functions for advanced text processing.

For more information about Tcl and SQL scripting capabilities for data replication, see the *Informatica Data Replication Scripting Guide*.

Database Character Set Conversion

To accurately replicate character data, verify character set settings for the source and target databases.

Data Replication can use the International Components for Unicode (ICU) library to convert character data from the source database encoding to the target database encoding. Data Replication supports character set conversion for configurations that have a DB2 for Linux, UNIX, and Windows, Microsoft SQL Server, MySQL, or Oracle source and an Amazon Redshift, Greenplum, Netezza, Vertica, or Teradata target. When you create a

replication configuration, the Data Replication Console queries the source and target databases to determine the source and target character sets and then writes these character set names to the replication configuration.

If the NLS_LANG environment variable is defined on the Oracle source, InitialSync uses this variable to determine the source database character set. If the NLS_LANG environment variable is not defined, InitialSync uses the character set from the replication configuration. The Applier always uses the source and target character sets from the replication configuration.

Virtual columns do not have the character set property. Instead, a virtual column uses the character set of the mapped source table to which you add the column. If the character set is not defined for the source table, the virtual column uses the character set that is defined for the source schema or database.

Restrictions:

- Data Replication does not support character set conversion for configurations that include Tcl scripts.
- Data Replication does not support character set conversion for constants that are used in SQL expressions.
- Data Replication does not support character set conversion for non-Latin characters in source database object names.

If the source character data includes only Latin characters but the source and target databases use incompatible character sets, Informatica recommends that you disable character set conversion to avoid performance degradation. To disable character set conversion, set the global.icu_enabled runtime parameter to 0. For example, disable character set conversion if the source character set is UTF-8 and the target character set is Latin 9.

Important: If the source character data includes non-Latin characters and the source and target databases use incompatible character sets, Data Replication ends with an error regardless of the global.icu_enabled setting.

For configurations that have sources other than DB2 for Linux, UNIX, and Windows, Microsoft SQL Server, MySQL, or Oracle and that have targets other than Amazon Redshift, Greenplum, Netezza, Vertica, or Teradata, the Extractor can convert source character data only from UTF-16 to UTF-8 encoding. For other source character encodings, the Extractor writes extracted change data to intermediate files in the original source character set. The Applier does not convert the character set of the change data when applying the data to the target. In this case, Data Replication requires the source and target databases to use the same character set.

For configurations with Oracle sources and Oracle targets, you can configure the Oracle target databases to convert the change data that the Applier and InitialSync load to the target character set. Define the NLS_LANG parameter on the systems where the Applier and InitialSync run. To accurately replicate data, set this parameter value to match the source database character set. In the Data Replication Console, create an environment variables list and add the NLS_LANG variable to it. Then assign this environment variables list to the Server Manager that runs the Applier or InitialSync. Oracle performs the conversion if the NLS_LANG environment variable value does not match the NLS_CHARACTERSET setting of the target database.

Replication of Database-Generated Values

Data Replication can replicate Insert and Update operations on source columns that contain database-generated values.

The following types of columns can contain database-generated values:

- DB2 for Linux, UNIX, and Windows identity columns

- Microsoft SQL Server identity columns
- MySQL identity columns
- Oracle columns that are the targets of numbers generated by sequence objects

Data Replication replicates the generated values from the source to the target.

If the target also contains identity columns or sequence objects, the following replication considerations apply:

- For Microsoft SQL Server targets, Data Replication uses the IDENTITY_INSERT statement to explicitly insert values into the identity columns on the target.
- For DB2 for Linux, UNIX, and Windows targets, you must use the GENERATED BY DEFAULT AS IDENTITY clause to define an identity column on the target. The Applier can then replicate generated values from the source to the identity column on the target. If you use the GENERATED ALWAYS AS IDENTITY clause, the Applier cannot replicate source generated values to the target identify column.
- If a failover to the target database occurs, you must manually reset the current value of an identity column or sequence object on the target prior to performing Insert and Update operations that retrieve generated values from the target. Execute a SELECT statement to determine the maximum value for the target column that holds the identity column or sequence object values. Then set the identity column or sequence object value to that maximum value plus an "increment by" value that is used to generate the next value.
  For example, a sequence object named GUID_SEQ, which has a last value of 100, is used to generate values for a column in an Oracle table. The maximum value for the table column is 224. To determine the next value using the "increment by" value of 124, execute the following SQL statements:

  ```
  ALTER SEQUENCE GUID_SEQ INCREMENT BY 124;
  SELECT GUID_SEQ.nextval FROM DUAL;
  ALTER SEQUENCE GUID_SEQ INCREMENT BY 1;
  ```
- Data Replication does not support bidirectional replication of database-generated values between columns that have a uniqueness constraint. To avoid uniqueness constraint violations, use conflict resolution rules.

Generated Virtual Indexes

Data Replication requires DB2, Microsoft SQL Server, and Oracle source databases to log additional information in expanded log records to replicate Updates, resolve conflicts, and filter replicated rows. When you save a configuration, the Data Replication Console creates virtual indexes for the primary key definition, conflict resolution rules, and filtering conditions.
Note: Data Replication does not generate or use virtual indexes, conflict resolution rules, and filtering conditions for MySQL sources.

To record generated virtual indexes, Data Replication enables the logging of additional information for a source in the following ways:

- For DB2 sources, Data Replication uses the DATA CAPTURE CHANGES option for all of the mapped source tables. This option causes DB2 to log full row images.
- For Microsoft SQL Server sources, Data Replication enables Change Data Capture for all of the mapped source tables.
- For Oracle sources, Data Replication enables supplemental logging for virtual index columns.

The Data Replication Console also enables additional logging for the virtual index columns or for the entire table, as appropriate for the source database. Data Replication generates index names based on the following criteria:

- For a primary key definition that is based on a database primary key, the Data Replication Console names the index by the database constraint name.
- For a primary key definition that includes all of the table columns, the index name consists of the table name followed by the _PK suffix.
- For conflict resolution rules, the index name consists of the table name followed by the _Conflict_Resolution suffix.
- For filtering conditions that you define on the **Map Columns** tab, the index name consists of the table name followed by the _WHERE suffix.
- For filtering conditions that you define on the **Routing** tab, the index name consists of the table name followed by the _FILTER suffix.

The following datatypes are not supported for additional logging because full before and after images are not available for them:

- For DB2 sources, BLOB, CLOB, DBCLOB, LOB, LONG VARCHAR, and LONG VARGRAPHIC datatypes
- For Microsoft SQL Server sources, IMAGE, NTEXT, NVARCHAR(-1), TEXT, XML, VARBINARY(-1), and VARCHAR(-1) datatypes
- For Oracle sources, BLOB, CLOB, LONG, LONG RAW, NCLOB, and RAW datatypes

Because these datatypes cannot be a part of a virtual index, you cannot use them to define a custom virtual index, a filtering condition, and conflict resolution rules.

Datatype Conversion Rules

The *DataReplication_installation*\uiconf\DataTypes.xml file defines rules that Data Replication uses to convert source datatypes to compatible target datatypes.

The conversion occurs when Data Replication generates target schema based on source schema or other target schema and when Data Replication replicates CREATE TABLE, ADD COLUMN, and MODIFY COLUMN changes.

The *DataReplication_installation*\uiconf\DataTypes.xml file contains a <ConversionRule> section for each combination of supported source and target types. The <ConversionRule> sections define the datatype conversion rules. You can edit the rules or add rules if necessary.

To understand how conversion rules are defined, review the following example <ConversionRule> sections in the DataTypes.xml file:

- **Example 1.** The following <ConversionRule> section defines a rule for converting the Oracle NUMBER datatype, which has a precision greater than 18 and a scale equal to 0, to the Netezza NUMERIC datatype with the same precision and scale:

```
<ConversionRule source_db="ORACLE" destination_db="NETEZZA">
  <DataTypes>
    <DataType>
      <OriginalDataType>NUMBER</OriginalDataType>
      <Condition><![CDATA[DataPrecision>18 && DataScale==0]]></Condition>
      <DataPrecision>DestinationDataPrecision=DataPrecision</DataPrecision>
      <DataScale>DestinationDataScale=DataScale</DataScale>
      <DestinationDataType>NUMERIC</DestinationDataType>
    </DataType>
```

```
  </DataTypes>
</ConversionRule>
```

- **Example 2.** The following <ConversionRule> section defines a rule for converting the Oracle VARCHAR datatype to the Teradata VARCHAR datatype:

```
<ConversionRule source_db="ORACLE" destination_db="TERADATA">
  </DataTypes>
    <DataType>
      <OriginalDataType>VARCHAR</OriginalDataType>
      <DestinationDataType>VARCHAR</DestinationDataType>
      <Calculation>DestinationDataSize=(DataSize+SrcMinCharsetSize-1)/
SrcMinCharsetSize*TgtCharsetSize</Calculation>
    </DataType>
  </DataTypes>
</ConversionRule>
```

  If the character sets of the source and target CHAR columns do not match, the target column size in bytes might differ from the source column size. In this case, Data Replication uses the following formula to calculate the target column size:

```
DestinationDataSize=(DataSize+SrcMinCharsetSize-1)/SrcMinCharsetSize*TgtCharsetSize
```

  SrcMinCharsetSize is the minimum number of bytes per character for the source character set. TgtCharsetSize is the number of bytes per character for the target character set. Data Replication gets the SrcMinCharsetSize and TgtCharsetSize values from the *DataReplication_installation*\uiconf\ICU_table.xml file.

  Important: Data Replication uses the formula that is defined by the Calculation parameter only if the character sets of the source and target columns do not match.

To add a conversion rule, add a <DataType> section under the <ConversionRule> section for the source and target type combination in the *DataReplication_installation*\uiconf\DataTypes.xml file. The <DataType> section has the following syntax:

```
<DataType>
  <OriginalDataType>source_column_datatype</OriginalDataType>
  <Condition>rule_condition<Condition>
  <DestinationDataType>target_column_datatype</DestinationDataType>
</DataType>
```

Replace the *source_column_datatype*, *rule_condition*, and *target_column_datatype* variables with the appropriate values.

Related Topics:

- "Generating Target Tables and Audit Log Tables" on page 198
- "CREATE TABLE and ADD COLUMN Operations" on page 55
- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Customizing Apply Settings for Target Tables" on page 225

DDL Replication

Data Replication supports the replication of many types of DDL operations from DB2 for Linux, UNIX, and Windows, Microsoft SQL Server, MySQL, and Oracle sources to a target.

In the Data Replication Console, you can select the types of DDL operations to replicate on the **Map Tables** and **Map Columns** tabs. Also, from the **Map Tables** tab, you can customize apply settings to indicate the types of DDL changes to apply.

When the Data Replication Console retrieves metadata for the configuration from the source database, the Console gets the current SCN, log coordinates, or LSN and writes this value to the replication configuration. When you run the Extractor later, it skips DDL changes that have SCN, log coordinate, or LSN values that are lower than the SCN, log coordinates, or LSN in the configuration.

When replicating DDL operations between heterogeneous sources and targets, the source and target databases might support different sets of DDL operations. You can replicate only the source DDL operations that can be mapped to compatible target DDL operations.

The Applier might skip a DDL operation or end abnormally if you replicate a source DDL operation that cannot be represented in the SQL syntax for the target type.

Related Topics:

- "DDL Replication Considerations" on page 54
- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Customizing Apply Settings for Target Tables" on page 225

DDL Replication Considerations

Review the following DDL replication considerations if you plan to enable DDL capture:

- Data Replication does not replicate DDL changes to Apache Kafka targets.
- Data Replication does not replicate DDL changes for bidirectional replication. For each of the two one-way configurations that you use for a bidirectional replication, Data Replication does not support unidirectional replication of DDL changes.
- If you try to edit a replication configuration for which DDL capture is enabled while a running replication task is processing a DDL change, the Data Replication Console displays a message that asks whether to stop the running replication tasks. If you edit the configuration while the tasks are processing a DDL change, you will not be able to save your configuration changes.
- Data Replication does not support replication of FOREIGN KEY and CHECK constraints.
- Data Replication does not support replication of RENAME TABLE operations.

  Note: If you rename tables on the source, Data Replication preserves the mappings of renamed tables and continues replication. When you open a configuration in the Data Replication Console, it detects renamed tables and updates the configuration.
- Data Replication does not support replication of the DEFAULT clause. When replicating a source DDL statement that uses the DEFAULT clause, the Applier executes the equivalent statement on the target without the DEFAULT clause.
  If a DDL statement uses both the DEFAULT clause and a NOT NULL constraint, the Applier executes the equivalent statement on the target without the DEFAULT clause and NOT NULL constraint to avoid NOT NULL constraint violations.
- For MySQL sources, Data Replication does not support DDL replication for tables that have columns with the JSON datatype or with spatial datatypes such as GEOMETRY.
- For Oracle 12*c* sources, Data Replication does not support replication of the IDENTITY clause in source DDL. When a source DDL statement contains the IDENTITY clause, the Applier executes the equivalent DDL statement on the target but without the IDENTITY clause.
- Data Replication does not support the replication of ALTER TABLE partition operations such as ADD PARTITION, SPLIT PARTITION, and DROP PARTITION.

- Data Replication does not replicate DROP COLUMN operations for the following types of columns that are involved in conflict resolution:
  - Columns for which source and target values are compared to detect Update conflicts
  - Columns that you use as resolution columns for the Maximum and Minimum resolution strategies
  - Columns for which you use before and after images as custom procedure parameters
- Data Replication does not replicate DDL operations that drop the last mapped table or column.
- To replicate DDL operations that change indexes or primary key constraints on the source to existing indexes and constraints on the target, ensure that the names of the target indexes and constraints match the names of the corresponding source indexes and constraints.
- When the Applier runs in continuous mode and a DDL change occurs on a DB2 source, the Applier applies the DDL change only after the Extractor processes the subsequent DML change record for any mapped source table.

Related Topics:

- "DDL Replication" on page 53
- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Customizing Apply Settings for Target Tables" on page 225
- "Supported DDL Operations for DB2 for Linux, UNIX, and Windows Sources" on page 65
- "Supported DDL Operations for Microsoft SQL Server Sources" on page 73
- "Supported DDL Operations for MySQL Sources" on page 77
- "Supported DDL Operations for Oracle Sources" on page 85
- "Supported DDL Operations for Amazon Redshift Targets" on page 88
- "Supported DDL Operations for DB2 for Linux, UNIX, and Windows Targets" on page 93
- "Supported DDL Operations for Greenplum Targets" on page 95
- "Supported DDL Operations for Microsoft SQL Server Targets" on page 97
- "Supported DDL Operations for MySQL Targets" on page 98
- "Supported DDL Operations for Netezza Targets" on page 101
- "Supported DDL Operations for Oracle Targets" on page 104
- "Supported DDL Operations for PostgreSQL Targets" on page 106
- "Supported DDL Operations for Teradata Targets" on page 110
- "Supported DDL Operations for Vertica Targets" on page 111

CREATE TABLE and ADD COLUMN Operations

If you enable the replication of CREATE TABLE or ADD COLUMN operations, when the Extractor captures these DDL changes, Data Replication adds the new table or column to the mapping in the configuration. Thereafter, Data Replication can replicate Insert, Update, and Delete operations for the added table or column.

To create a table or a column, Data Replication uses the datatype conversion rules that are defined in the <ConversionRules> section of the *DataReplication_installation*\uiconf\DataTypes.xml file.

For CREATE TABLE operations, Data Replication creates the target table and maps it to the source table. The following considerations apply to CREATE TABLE processing:

- Data Replication processing of CREATE TABLE operations depends on the target type in the following manner:
  - For Amazon Redshift, Greenplum, Netezza, Teradata, and Vertica targets, Data Replication creates the target table and a corresponding audit log table and then maps the source table to the target table for Merge Apply processing.
  - For DB2, Microsoft SQL Server , MySQL, PostgreSQL, and Oracle targets, Data Replication creates a target table and then maps the source table to the target table for SQL Apply processing.
  - For Cloudera, Flat File, and Hortonworks targets, Data Replication creates flat files that correspond to the newly created source table after one or more records are inserted into the new table.
- If you replicate CREATE TABLE operations from Microsoft SQL Server sources, manually enable Change Data Capture for the source tables. Otherwise, Data Replication cannot replicate change data to the target tables that the CREATE TABLE operations created.

Data Replication processing of ADD COLUMN operations depends on the apply mode in the following manner:

- In Audit Apply mode, Data Replication creates the columns for before and after values, which correspond to the added column, in the audit log table, if the table exists.
- In Merge Apply mode, Data Replication creates the target column and also creates the corresponding audit log table columns for before and after values in the target database.
- In SQL Apply mode, Data Replication creates the target column.

Related Topics:

- "Datatype Conversion Rules" on page 52
- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Customizing Apply Settings for Target Tables" on page 225

Logging Supplemental Information for Source Tables Created with Replicated DDL

If you replicate CREATE TABLE operations, you must enable supplemental logging for the new tables to store the additional information that is required for data replication in the source database log files.

You can enable supplemental logging from the Data Replication Console, with Server Manager CLI commands, or in the source database.

Enable supplemental logging before allowing DML changes to be written to the source tables. If you enable supplemental logging after DML changes are written to the tables, the Extractor extracts incomplete change data and the Applier cannot process Updates and Deletes correctly.

In the Data Replication Console, add the new table to the mapping in the replication configuration. When you save the configuration, the Console can enable supplemental logging for the table provided that you set the **Supplemental Logging Settings** field to **Ask** or **Yes** on the **Runtime Settings** tab > **General** view. For Oracle sources, the Data Replication Console generates a supplemental log group for the table if the table has a primary key or unique index. Ensure that the supplemental log group includes the primary key or unique index columns. The generated supplemental log group names have the format IDR_*table_object_ID_sequence_number*.

Note: Because Oracle restricts the number of columns in a supplemental log group to 33, the Data Replication Console might generate more than one supplemental log group for a table. For example, a source

table with 100 columns has four supplemental log groups: three groups with 33 columns each and one group with one column.

To manually enable or manage supplemental logging for source tables from the Data Replication Console, click the **Manage Database Supplemental Logging** button on the **Map Tables** tab. Depending on the source type, perform one of the following actions:

- For DB2 sources, select **Data Capture** for each new source table to set the DB2 DATA CAPTURE CHANGES option.
- For Microsoft SQL Server sources, select **CDC** for each new source table to set SQL Server Change Data Capture.
- For Oracle sources, select the columns to be included the supplemental log group for each table. If a new Oracle source table does not have primary key or unique index and you did not choose to create a virtual index when you saved the configuration, include all of the table columns in the supplemental log group except those that have the following datatypes:

  - BLOB
  - CLOB
  - LONG
  - LONG RAW
  - NCLOB
  - RAW
  - Any of the Oracle datatypes that Data Replication does not support for replication

  Important: If you create a unique index on a new target table, also add the source columns that correspond to the target index columns to the supplemental log group for the source table. Otherwise, Data Replication might not correctly apply Updates to the unique index columns.

If you want to enable supplemental logging for the new tables from the source database, execute the following SQL statements immediately after creating the source tables:

- For Oracle sources, use `ALTER TABLE table_name ADD SUPPLEMENTAL LOG GROUP IDR_supplental_log_group_name (column_list) ALWAYS;`
- For DB2 sources, use `db2 alter table table_name data capture changes`
- For Microsoft SQL Server sources, use `execute sp_cdc_enable_table 'schema_name', 'table_name', 'cdc_capture_instance_name', null, null, null, null, null, 1`

For more information about managing supplemental logging, see "Managing Database Supplemental Logging" on page 342 or the *Data Replication Command Line Interface for the Server Manager*.

Replication of TRUNCATE TABLE Operations

If you enable replication of TRUNCATE TABLE operations, Data Replication can process TRUNCATE TABLE operations for most source types. How Data Replication processes source TRUNCATE TABLE operations on the target depends on the apply mode and the target database type.

The apply mode affects TRUNCATE TABLE apply processing in the following manner:

- In SQL Apply and Audit Apply modes, Data Replication truncates the mapped target table or audit log table.

  Important: Do not enable replication of TRUNCATE TABLE operations in Audit Apply mode if you want to preserve the change data in the audit log tables.
- In Merge Apply mode, Data Replication truncates the mapped target table but does not truncate the corresponding audit log table.

Additionally, some target types have special apply considerations:

- For Netezza and MySQL targets, Data Replication replicates TRUNCATE TABLE operations as DELETE FROM operations.
- For Teradata targets, Data Replication replicates TRUNCATE TABLE operations as DELETE ALL operations.
- For Apache Kafka, Cloudera, Flat File, and Hortonworks targets, Data Replication does not support TRUNCATE TABLE operations. Do not enable replication of TRUNCATE TABLE operations for these targets.

Related Topics:

- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Customizing Apply Settings for Target Tables" on page 225

Order in Which Data Replication Applies DDL Operations

If you replicate source DDL and DML operations within one apply cycle, the Applier applies all of the DDL changes prior to applying the DML changes.

The following cases pertain to the order in which DDL operations are applied within an apply cycle:

- If you update a NULL value in the column to set a value other than NULL and then run an ALTER COLUMN statement that adds a NOT NULL constraint to this column in the same replication cycle, the Applier fails.

  For example, replicate the following source statements in the first cycle:

  ```
  CREATE TABLE test (col1 NUMBER NOT NULL, col2 NUMBER NULL);
  INSERT INTO test (col1, col2) VALUES (1, NULL);
  COMMIT;
  ```

  Then replicate the following statements in the second cycle:

  ```
  UPDATE tst SET col2 = 0 WHERE col1 = 1;
  COMMIT;
  ALTER TABLE tst MODIFY (col2 NOT NULL);
  ```

  In the second cycle, the Applier first replicates the ALTER TABLE operation but fails to set the NOT NULL constraint for the column that stores the NULL value.
- If you insert data into a source table and then modify the datatype of a table column in the same apply cycle, the Applier processes the column datatype change prior to applying the Inserts to the target. When the Applier applies the Inserts to the target, it does not insert data into the column with the modified datatype.
- If you make DML changes to a source table and then drop this table in the same apply cycle, the Applier processes the DROP TABLE operation and removes the mapping from the configuration first. Then the Applier skips the DML changes that were captured for this table.

Examples of DDL Replication

Review the following examples to learn how to replicate DDL operations from a source to a target. These examples demonstrate the replication of basic DDL operations in SQL Apply, Audit Apply, and Merge Apply modes.

- **Example 1**. Replicate a CREATE TABLE operation from a Microsoft SQL Server source to an Oracle target in SQL Apply mode.

The following DDL operation occurs on the Microsoft SQL Server source:

```
CREATE TABLE [dbo].[TABLE_DEMO](
    [IDKEY] [int] NOT NULL PRIMARY KEY,
    [DATA] [varchar](20) NULL
);
```

Because you enabled the replication of CREATE TABLE operations for the "dbo" SQL Server owner and the "ORA_TARGET" Oracle schema, Data Replication executes the following statement on the Oracle target to create an identical Oracle table with the schema name "ORA_TARGET":

```
CREATE TABLE "ORA_TARGET"."TABLE_DEMO"
(
 "IDKEY" NUMBER (10, 0)  NOT NULL,
 "DATA" VARCHAR2(20) NULL,
 PRIMARY KEY ("IDKEY")
)
```

- **Example 2**. Replicate an ADD COLUMN operation from a Microsoft SQL Server source to an Oracle target in Audit Apply mode.

  The following DDL operation adds the INTDATA column to the TABLE_DEMO table on the Microsoft SQL Server source:

  ```
  ALTER TABLE [dbo].[TABLE_DEMO] ADD [INTDATA] INTEGER;
  ```

  Because you enabled replication of ADD COLUMN operations for the TABLE_DEMO source table, Data Replication creates two columns in the corresponding audit log table on the target for the before and after values in the INTDATA column. Data Replication executes the following statement on the Oracle target:

  ```
  ALTER TABLE "ORA_TARGET"."TABLE_DEMO_LOG" ADD ("INTDATA_OLD" NUMBER(10) NULL ,
  "INTDATA_NEW" NUMBER(10) NULL )
  ```

- **Example 3**. Replicate a CREATE TABLE operation from an Oracle source to a Netezza target in Merge Apply mode.

  The following DDL operation occurs on the Oracle source:

  ```
  CREATE TABLE ORA_SOURCE.TABLE_DEMO
  (
   IDKEY NUMBER(22,0) NOT NULL PRIMARY KEY,
   DATA VARCHAR2(20)
  );
  ```

  Because you enabled replication of CREATE TABLE operations for the "ORA_SOURCE" source schema and the "NZ_TARGET" target schema, Data Replication creates a corresponding target table and audit log table on the target. Data Replication executes the following statements on the Netezza target:

  ```
  CREATE TABLE "NZ_TARGET"."TABLE_DEMO"
  (
   "IDKEY" NUMERIC(22,0) NOT NULL,
   "DATA" VARCHAR(20) NULL
  )
  CREATE TABLE "NZ_TARGET"."TABLE_DEMO_LOG"
  (
   "OP_TIME" TIMESTAMP ,
   "OP_CODE" CHAR(1) NOT NULL,
   "OP_CMT_SCN" NUMERIC(20) NOT NULL,
   "OP_CMT_TIME" TIMESTAMP ,
   "OP_XID" NUMERIC(22) ,
   "OP_NUM_IN_TX" BIGINT NOT NULL,
   "OP_KEY_LEVEL" INTEGER ,
   "OP_ROOT_KEY_ROWID" BIGINT ,
   "IDKEY_OLD" NUMERIC(22,0) NOT NULL,
   "IDKEY_NEW" NUMERIC(22,0) NOT NULL,
   "DATA_OLD" VARCHAR(20) NULL,
   "DATA_NEW" VARCHAR(20) NULL
  )
  ```

  Because Netezza does not use indexes, Data Replication ignores the primary key definition in the source table.

CHAPTER 3

Sources - Preparation and Replication Considerations

This chapter includes the following topics:

DB2 for Linux, UNIX, and Windows Sources

Preparing DB2 for Linux, UNIX, and Windows Source Systems

To prepare a DB2 for Linux, UNIX, and Windows source system for data replication, use the appropriate DB2 commands to complete the following steps.

1. Verify that Version 10.5 of either the DB2 Client or DB2 Connect is installed.

   Important: If the Extractor task extracts data from a remote DB2 database server, the DB2 Fix Pack version of the DB2 Client or DB2 Connect must be equal to or later than the DB2 Fix Pack version of the DB2 database.
2. Back up the DB2 database before you configure it for data replication.
3. Set the overflowlogpath parameter to specify the location that each DB2 source database searches for the log files that the db2Readlog API needs to access. Use the following command:

   ```
   db2 update db cfg for database_name using overflowlogpath overflow_log_path
   ```

   Note: The Extractor interacts with the db2Readlog API to extract data for replication.
4. Set the logarchmeth1 or logarchmeth2 parameter to specify the directory in which DB2 stores the archive log files. Use the following command:

   ```
   db2 update db cfg for database_name using {logarchmeth1|logarchmeth2}
   DISK:archive_log_path
   ```

   Note: Setting a log archive method turns on archive logging in the database.

5. If you plan to perform bidirectional replication or cascade replication, or if you want to view additional information about DB2 transactions in the **Transactions of configuration <configuration_name>** dialog box in the Data Replication Console, perform one of the following actions:

   - If you use DB2 9.7, run the following command to set the DB2_LOGGING_DETAIL registry variable for the source database instance to APPLINFO:

     ```
     db2set -i <varname>instance_name</varname> DB2_LOGGING_DETAIL=APPLINFO
     ```

   - If you use DB2 10.1 or later, run the following command to set the log_appl_info parameter for the source database to yes:

     ```
     db2 update db cfg for database_name using log_appl_info yes
     ```

   These commands support loopback avoidance by causing DB2 to write an additional log record that contains information about DB2 transactions. If you skip this step, the Server Manager cannot retrieve the following information for DB2 transactions: database transaction status, application ID, database user name, system user, and host. Also, the Extractor cannot distinguish the transactions that originally occurred on the database from the transactions that the Applier applied. As a result, the Extractor captures all of the change data.

   Skip this step if you do not plan to use bidirectional or cascade replication and if you do not need to view the additional information about DB2 transactions in the Data Replication Console.

6. If you changed the DB2_LOGGING_DETAIL registry variable or log_appl_info parameter in step 5, restart the database instance so that the new parameter settings take effect. Use the following commands:

   ```
   db2stop
   db2start
   ```

7. Back up the DB2 database before you start data replication.

8. Archive the current database log. Use the following command:

   ```
   db2 archive log for database database_name
   ```

9. Ensure that the DATA CAPTURE CHANGES option is enabled for the source tables for which you want to replicate data in one of the following ways:

   - If you plan to define replication configurations and run InitialSync and the Extractor from the Data Replication Console, when you save a configuration in the Console later, the Console will prompt you to enable DATA CAPTURE CHANGES. Click **Yes** to enable this option.
   - If you plan to run InitialSync and the Extractor outside of the Data Replication Console, manually run the following command for each source table:

     ```
     db2 alter table table_name data capture changes
     ```

     If the source tables include LONG VARCHAR or LONG VARGRAPHIC columns, use the following command:

     ```
     db2 alter table table_name data capture changes include longvar columns
     ```

   Note: After you create and save a new configuration, you can manage the DATA CAPTURE settings for the source tables on the **Map Tables** tab. For more information, see "Managing DB2 DATA CAPTURE Settings" on page 342.

10. Grant DBADM authority to the user ID that Data Replication uses to connect to the DB2 source. Use the following command:

    ```
    db2 grant dbadm on database to user datarep_user
    ```

11. Ensure that the DB2INSTANCE environment variable is set to the DB2 Client instance name on the system where the Extractor runs.

12. Ensure that the library path environment variable includes the directory that contains the DB2 client libraries. Use one of the following library path environment variables, depending on the type of operating system where the Extractor runs:

    - LD_LIBRARY_PATH on Linux
    - LD_LIBRARY_PATH_64 on Solaris
    - LIBPATH on AIX
    - PATH on Windows
    - SHLIB_PATH on HP-UX

    For example, use the following command on Linux:

    ```
    export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/db2inst1/sqllib/lib32:/home/db2inst1/
    sqllib/lib64
    ```

13. For DB2 sources that use accented and non-Latin characters, set the DB2CODEPAGE environment variable to an encoding that is compatible with the source character set on the computer where InitialSync runs. For example, set DB2CODEPAGE to 1208 for UTF-8 encoding.
14. If you want to connect to a remote DB2 database by using a database alias, perform the following substeps:

    a. To provide TCP/IP access to a remote DB2 server node that was not previously cataloged, issue the following DB2 catalog tcpip node command on the computer where the Extractor runs:

    ```
    db2 catalog tcpip node new_node_name remote source_server_ip server
    source_db2_port
    ```

    The *new_node_name* variable value must be unique in the DB2 node catalog.

    Note: The default DB2 port number is 50000.

    To issue this command, you must have sysadm or sysctrl authority.

    b. To store the location of a DB2 database on a remote node that was not previously cataloged in the system database directory, issue the following DB2 catalog database command on the computer where the Extractor runs:

    ```
    db2 catalog database source_db2_database as new_db2_database_alias at node
    new_node_name authentication server
    ```

    The *new_db2_database_alias* variable value must be unique in the DB2 database catalog.

    To issue this command, you must have sysadm or sysctrl authority.

15. Add the operating system user that runs the Extractor to a user group with DB2 SYSADM authority.

    a. Determine the SYSADM group name:

    - On Linux and UNIX, use the following command:

    ```
    db2 get dbm cfg | grep -i SYSADM_GROUP
    ```

    - On Windows, use the following command:

    ```
    db2 get dbm cfg | findstr SYSADM_GROUP
    ```

    Important: If the SYSADM_GROUP parameter is not defined, DB2 sets SYSADM authority to the following groups and accounts:

    - On Linux and UNIX, SYSADM authority is granted to the primary group of the instance owner.
    - On Windows, SYSADM authority is granted to the local Administrators group, Administrators group, DB2ADMNS group, and LocalSystem account.

    b. Add the operating system user to the group that is specified by the SYSADM_GROUP parameter.

Replication Considerations for DB2 for Linux, UNIX, and Windows Sources

Review the following information about Data Replication support for DB2 for Linux, UNIX, and Windows sources:

- Data Replication can extract change data from the following types of DB2 tables:
  - DB2 materialized query tables
  - DB2 insert time clustering (ITC) tables
  - DB2 multidimensional clustering (MDC) tables
  - DB2 range-clustered tables (RCTs)
  - DB2 partitioned tables
  - DB2 source tables that use value compression
  - DB2 source tables that use row compression
- If the source tables use row compression, ensure that the extract.db2.enable_row_compression_support runtime parameter is set to 1. When you create a replication configuration that maps a source table that uses row compression, the Data Replication Console sets this parameter to 1. The following guidelines apply to using the parameter setting of 1:
  - The parameter setting of 1 might degrade replication performance. Use this setting only if one or more mapped source tables use row compression.
  - To prevent data loss, set the parameter to 1 before you enable row compression for a mapped source table.
  - To prevent data loss, set the parameter to 1 if you enable DDL replication and new tables might use row compression.
- On Linux and Windows, Data Replication can capture data from DB2 online and archive logs.
- Data Replication does not support source tables in a partitioned database environment that was created with the DB2 Database Partitioning Feature (DPF). However, Data Replication does support target tables in this type of environment.
- The DB2 DATA CAPTURE CHANGES option must be set for the DB2 source tables.
- Archive logging must be active for the source database.
- If you do not override the default Start Point value for the Extractor, you must run the Extractor prior to starting InitialSync. This initial Extractor run writes an LSN value that is lower than the synchronization transaction LSN, to the db2.initial_lsn parameter in the configuration file. After the InitialSync task completes, when you start the Extractor again, the Extractor uses the LSN in the configuration file to determine the point from which to start reading change data from the transaction logs.
- Data Replication does not support the replication of triggers and stored procedures.
- Data Replication does not support DB2 pureScale.
- If you plan to connect to a remote DB2 source by using a database alias, specify the alias name in the extract.db2.db_alias_name advanced runtime parameter. The Extractor then uses the database alias connection string instead of the connection details that are defined on the **Source Database** tab. InitialSync always uses the connection information that is defined on the **Source Database** tab.

  Tip: Use the following command to list DB2 aliases on the system where the Extractor runs:

  ```
  db2 list database directory
  ```
- If the source tables are compressed, ensure that you have a compression dictionary that is compatible with the compressed DB2 log records from which Data Replication reads change data for the tables. Otherwise, DB2 cannot decompress the log records for the Extractor read requests. Usually, the

compatible compression dictionary is available because DB2 maintains the current compression dictionary and a backup of the previous compression dictionary on disk.
If you run the DB2 REORG TABLE utility or the DB2 LOAD utility with the REPLACE or RESUME NO option against compressed source tables, Informatica recommends that you specify the KEEPDICTIONARY option for the utility. The KEEPDICTIONARY option forces DB2 to retain the current compression dictionary if it exists. If you use the RESETDICTIONARY option, DB2 rebuilds the compression dictionary. In this case, the previous compression dictionary that matches the DB2 log records might not be available any longer.

Tip: If you must use the RESETDICTIONARY option, run the Extractor prior to running the utility to process all of the new records.

- If you plan to connect to multiple DB2 source instances that have different versions, use an environment variables list that contains the DB2INSTANCE and library path environment variables for each DB2 version. You can add custom environment variables to use on a Server Manager system for a particular replication configuration. For more information about environment variables lists, see "Configuring an Environment Variables List for the Main Server" on page 130.

  Important: The library path environment variable must specify a single DB2 client directory. If you specify multiple DB2 client directories, the Extractor uses the first path from this environment variable.

- Ensure that the Extractor has processed all of the change data records before you change any of the following log file parameters for a DB2 source that is already used in replication configurations:

  - Logfilsiz
  - Logprimary
  - Logsecond
  - Newlogpath

  If you edit any of these parameters before the Extractor finishes processing the change data records, the changes in the log structure might cause data loss.

- If Extractor performance is significantly degraded and you use DB2 automatic statistics collection or automatic maintenance, try disabling some of these DB2 automatic operations. For more information, contact your DB2 DBA.
- After you upgrade a DB2 for Linux, UNIX, and Windows source database to DB2 Version 10.5 from an earlier DB2 version, the Extractor might not be able to process the database transaction logs that were created prior to the upgrade. To avoid this problem, follow the procedure that is described in "Resuming Replication After Upgrading a DB2 for Linux, UNIX, and Windows Source Database" on page 373.

Related Topics:

- "Database Objects That Cannot Be Mapped" on page 219

DBSYNC_SYNC_LSN Table

For DB2 for Linux, UNIX, and Windows sources, Data Replication uses a service table, which has the default name of DBSYNC_SYNC_LSN, to determine the Sync Point (LSN) value and the current LSN value at the time the Data Replication Console retrieves metadata for the configuration.

Note: For DB2 for Linux, UNIX, and Windows sources, the Sync Point value is an LSN.

To determine the Sync Point values, Data Replication uses the records that InitialSync adds to the service table for each synchronized table. The Extractor writes the Sync Point value to the configuration for each synchronized table.

To determine the current LSN value, Data Replication uses the records that the Data Replication Console adds to the service table when refreshing schema and table metadata. When you run the Extractor later, it uses these records to determine the point in the transaction log from which to start processing DDL changes.

After you save a replication configuration, Data Replication deletes the records for the configuration that have an older session ID. Thereafter, the DBSYNC_SYNC_LSN table contains records for the latest session ID.

The following table describes the columns in the DBSYNC_SYNC_LSN table:

Column	Description
IDNTT	An incremental numeric identifier for the record.
CONFIG_NAME	A replication configuration name.
SCHEMA_NAME	A source schema name.
TABLE_NAME	A source table name.
APP_TYPE	The record type. Valid values are: - **S**. Indicates a schema LSN record. The Data Replication Console inserts a record of this type when getting metadata for the specified source schema. - **T**. Indicates a table LSN record. The Data Replication Console inserts a record of this type when getting metadata for the specified table. - **F**. Indicates a final LSN record. The Data Replication Console inserts a record of this type after saving the specified configuration. - **I**. Indicates the Sync Point (LSN) record that InitialSync adds after synchronizing the target table with the source table.
USER_ID	A unique user ID for the Data Replication user who opened, created, refreshed, or saved the specified replication configuration.
SESSION_ID	A unique identifier for the session. A session begins when you create, open, or refresh a configuration. A session ends when you save the configuration.

Related Topics:

- "Sync Point Value" on page 272
- "Applier Handling of the Sync Point Value" on page 37

Supported DDL Operations for DB2 for Linux, UNIX, and Windows Sources

The Extractor for DB2 for Linux, UNIX, and Windows sources can process the following DDL operations:

- `ALTER TABLE table_name ADD column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name {PRIMARY KEY|UNIQUE} (column_name)`
- `ALTER TABLE table_name ALTER COLUMN column_name SET DATA TYPE datatype`
- `ALTER TABLE table_name ALTER COLUMN column_name SET NOT NULL`
- `ALTER TABLE table_name ALTER column_name DROP NOT NULL`
- `ALTER TABLE table_name DROP column_name`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name`
- `CREATE [UNIQUE] INDEX index_name ON table_name`

- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL] [PRIMARY KEY] [UNIQUE])`
- `CREATE TABLE table_name AS (SELECT statement)`
- `CREATE TABLE table_name LIKE table_name`
- `DROP INDEX index_name`
- `DROP TABLE table_name`
- `TRUNCATE TABLE table_name`

Notes:

- The Applier processes the CREATE INDEX statement only if this statement creates an index in the same schema as the parent table.
- When a CREATE TABLE AS (SELECT...) operation occurs on the source, Data Replication creates a target table using the appropriate CREATE TABLE statement. After you execute REFRESH statements for the source table, Data Replication begins applying DML operations to the target table.

Microsoft SQL Server Sources

Preparing Microsoft SQL Server Source Systems

After you install Informatica Data Replication on a Microsoft SQL Server source system, you must complete several tasks to prepare the system for data replication.

1. Download and install .NET Framework 3.5 SP1 or later from the Microsoft web site if it is not currently installed.
2. Verify that the Microsoft SQL Server database uses Mixed Mode Authentication to allow use of SQL Server Authentication.
3. Create an SQL Server login account that uses SQL Server Authentication. Use SQL Server Management Studio or the following Transact-SQL statement:

   ```
   CREATE LOGIN datarep_user WITH PASSWORD = 'password';
   GO
   ```

4. Create a user in the master database and grant the permissions that Data Replication requires to this user. Use the following SQL statements:

   ```
   use [master]
   CREATE USER datarep_user FOR LOGIN datarep_user;
   GRANT VIEW SERVER STATE TO datarep_user;
   GRANT VIEW ANY DEFINITION TO datarep_user;
   GRANT SELECT TO datarep_user;
   GO
   ```

5. Create a user for each source database and assign the db_owner role to this user. Use the following SQL statements:

   ```
   use [source_mssql_database_name]
   CREATE USER datarep_user FOR LOGIN datarep_user;
   EXEC sp_addrolemember 'db_owner', 'datarep_user'
   GO
   ```

   Data Replication requires the db_owner role to perform the following tasks:

   - To connect to Microsoft SQL Server sources from the Data Replication Console

- To create backup logs for Microsoft SQL Server sources with the Server Manager
- To enable Change Data Capture for a replication job

Note: Data Replication also requires the "public" server role, to which every database user belongs by default.

6. To use the Server Manager to create backup logs in SQL Server native format, assign the db_backupoperator role to the user in each source database. Use the following SQL statements:

```
use [source_mssql_database_name]
EXEC sp_addrolemember 'db_backupoperator', 'datarep_user'
GO
```

Note: Data Replication can capture change data from backup logs in SQL Server native format.

7. If you plan to run the Server Manager on a computer that is remote from the SQL Server source, set the **remote admin connections** option of the SQL Server sp_configure stored procedure to 1. Use the following commands:

```
sp_configure 'remote admin connections', 1;
GO
RECONFIGURE;
GO
```

This setting enables the remote Server Manager to use a dedicated administrator connection (DAC) to the SQL Server source.

8. Optionally, if you capture data from online transaction logs, assign the sysadmin role to the user for each source database to enhance performance. You can run the following stored procedure:

```
use [source_mssql_database_name]
EXEC sp_addsrvrolemember @loginame= 'datarep_user', @rolename = 'sysadmin'
GO
```

The Extractor task uses one of the following methods to get the list of active virtual log files (VLFs) from the Microsoft SQL Server source:

- If the user has the sysadmin role, the Extractor uses the DBCC LOGINFO command to get the information about the VLFs in the transaction log. This method is faster.
- If the user does not have the sysadmin role, the Extractor scans transaction logs and processes VLF headers to get the information about the VLFs in the transaction log without running the DBCC LOGINFO command. This method is slower.

9. Ensure that all databases from which data will be extracted are set to the Full Recovery Model.
10. Create a full backup of each database from which data will be extracted.
11. Enable snapshot isolation for all of the SQL Server source databases to correctly replicate changes that might occur during InitialSync initial synchronization of the target tables with the SQL Server source tables. Use the following syntax to run the ALLOW_SNAPSHOT_ISOLATION statement for each of the source databases:

```
ALTER DATABASE source_mssql_database_name SET ALLOW_SNAPSHOT_ISOLATION ON;
```

Note: You can skip this step if you do not use InitialSync for initial synchronization of the source and target tables.

12. In Microsoft SQL Server Configuration Manager, under Network Configuration, enable TCP/IP as the protocol for the SQL Server instance.
13. For Microsoft SQL Server 64-bit software, add `DBSYNC_HOME\support` to the PATH environment variable.

Replication Considerations for Microsoft SQL Server Sources

Review the following information about Data Replication support for Microsoft SQL Server sources:

- The Data Replication Extractor reads data directly from SQL Server transaction logs. It does not use the Microsoft SQL Server API.
- For Data Replication to capture data from SQL Server logs, SQL Server Change Data Capture must be enabled for all of the mapped source tables.
- Data Replication supports Microsoft SQL Server tables that do not have primary keys.
- Data Replication can extract change data from tables that use row compression. However, Data Replication does not extract change data from tables that use page compression.
- Data Replication can extract change data from sparse columns.
- Data Replication does not extract change data from views.
- To capture data from the online transaction logs, the Extractor must run on the same Windows system as the SQL Server database. Also, the user account for the SQL Server database must be a Windows Administrator account that uses Windows Authentication.

  Important: On the SQL Server source system, you must run the Server Manager and the Extractor that extracts SQL Server changes under a Windows Administrator user account.
- Data Replication can capture change data from backup transaction logs that are in SQL Server native format and that not compressed. To create backup logs in native format, you can run a Microsoft SQL Server Backup task under a Server Manager in the Data Replication Console or use a third-party tool. If you use a third-party tool, ensure that the backup logs are not compressed. Data Replication does not capture change data from compressed backup logs, even if they are in native format.

  Important: If you use a third-party tool to create backup logs, each time you create the backup logs, use a different media set and unique backup log file names. If you append backup logs to an existing media set or overwrite an existing backup log, the Extractor might process change data from the backup logs incorrectly.
- To capture data from backup logs, you can run the Extractor and the Server Manager instance either on the SQL Server source system or on another system, which can have a different operating system.

  If you run the Extractor and Server Manager on a system other than the SQL Server source system, you can make the backup logs available to the Extractor in one of the following ways:

  - If you run the Extractor and Server Manager instance on the source system, perform one of the following tasks:
    - Send the backup logs to the Server Manager on the system where the Extractor runs by using a Copy File task.
    - Schedule a Microsoft SQL Server Backup task to create backup logs in a shared directory that the Extractor can access over the network. The Extractor can then read the generated backup logs from the shared directory.
  - If you do not run the Extractor and Server Manager instance on the source system, schedule SQL Server to create backup logs in a shared directory that the Extractor can access.

  Important: For the Extractor to read data from backup logs, ensure that you clear the **Read from online transaction logs** option on the **Extract Range** tab.
- Data Replication can extract change data from an SQL Server database while SQL Server native transactional replication is active. In the Data Replication Console, on the **Runtime Settings** tab > **General** view, select the **Disable secondary truncation checkpoint** option. This option prevents conflicts between Data Replication and SQL Server truncation checkpoints in the transaction logs by causing Data Replication to use the SQL Server truncation checkpoints.

- SQL Server 2008 R2 logs Update operations as a pair of Delete and Insert operations. By default, the Extractor tries to merge the Delete and Insert operations into a single Update operation and write that Update to the intermediate files. This Extractor behavior improves the performance of Applier processing of Updates in SQL Apply mode.

  If you use SQL Server 2008 R2 and either Merge Apply or Audit Apply mode to replicate change data, set the extract.mssql.process_updates_as_updates parameter to 0 to extract each Update as a pair of Delete and Insert operations.

  Note: Later supported SQL Server versions process Updates as Updates.
- If you use the default value of 1 for the extract.mssql.process_updates_as_updates runtime parameter, Microsoft SQL Server does not log before images of LOB columns for Delete operations. Consequently, when you replicate LOB data from a Microsoft SQL source to flat file targets or target databases in Audit Apply or Merge Apply mode, the flat file rows or audit log table rows for the Delete operations contain null values in the before- and after-image columns that correspond to the source LOB columns.
- Data Replication does not support virtual computed columns. If you map a virtual computed column to a target column, the target column receives null values during replication processing.
- Data Replication does not support SQL Server database encryption.
- Data Replication does not support the replication of triggers and stored procedures.
- If you extract change data from online transaction logs, you must stop replication processing before using the DBCC SHRINKDATABASE or DBCC SHRINKFILE command to shrink the size of log files.
- If you run the Extractor against SQL Server online transaction logs in Continuous mode, the Extractor starts each of its microcycles by issuing a SQL Server DBCC LOGINFO command. This command gets information about each virtual log file (VLF), such as the file size, offset, and sequence number. If the online transaction log is large and contains many VLFs, this processing can increase CPU usage and the workload on the source database, particularly under the following conditions:
  - You use the continuous replication latency of 0.333 second or less, which causes the Extractor to start another microcycle after a minimal wait interval.
  - The source database is idle, which causes the Extractor to re-read the same VLFs each microcycle.

  To reduce the CPU usage and database workload, increase the value of the **Continuous replication latency** field on **Runtime Settings** tab > **General** view.
- Data Replication does not support the following index types:
  - Spacial indexes
  - XML indexes
  - Full-text indexes

  Data Replication does not capture DDL changes to these indexes and cannot use them for accurate replication of Updates and Deletes.
- Data Replication supports Transparent Data Encryption (TDE) of Microsoft SQL Server source databases. The Extractor can decrypt the data in the database log files that have been encrypted with either the Advanced Encryption Standard (AES) or Triple DES (3DES) encryption algorithm.

  To replicate change data from encrypted databases, you must first specify the required certificates in the replication configuration. For more information, see "Enabling Replication from Microsoft SQL Server Databases That Use Transparent Data Encryption" on page 258.

  When the Extractor loads the replication configuration, it loads all of the TDE certificates that were added to the replication configuration. The Extractor uses the certificates to decrypt the database encryption keys from virtual log file (VFL) headers. The Extractor then uses the keys to decrypt change data from the encrypted databases.

The Extractor writes decrypted data to intermediate files. After the intermediate files are transmitted to the target, the Applier reads the decrypted values from the files and writes these values to the target.

Important: The decryption of data from encrypted Microsoft SQL Server databases can significantly degrade Extractor performance.

- Data Replication can extract change data from Microsoft SQL Server sources that use Always On Availability Groups if the availability replicas in the group use synchronous-commit mode.

  Restriction: Data Replication does not support availability replicas that use asynchronous-commit mode.

  If you extract change data from backup transaction logs, you must add backup log locations on the **Extract Range** tab for each availability replica. If you specify shared locations that might become temporarily unavailable, set the extract.mssql.ignore_inactive_backup_locations runtime parameter to 1 to ensure that the Extractor does not end with an error if it cannot access a backup log location.
- Data Replication does not support Microsoft SQL Server source databases that use the Always Encrypted option.

Related Topics:

- "Editing Microsoft SQL Server Instance Settings" on page 135
- "Supported DDL Operations for Microsoft SQL Server Sources" on page 73
- "Database Objects That Cannot Be Mapped" on page 219

Coordination with Microsoft SQL Server Change Data Capture

To capture change data from Microsoft SQL Server transaction logs, you must enable SQL Server Change Data Capture for the SQL Server source databases and tables. Native Microsoft SQL Server replication also uses Change Data Capture.

When SQL Server Change Data Capture is enabled, Data Replication leverages the Microsoft SQL Server replication stored procedures for change data replication. These procedures perform the following functions:

- Force Microsoft SQL Server to log Updates as a pair of Delete and Insert operations. This type of logging makes the before and after images of Update operations available to the Extractor. The Extractor can then accurately replicate the Updates.
- Force Microsoft SQL Server to log additional information that the Extractor uses to determine the points in the log at which LOB data starts and ends. The Extractor can then correctly replicate LOB columns.
- Mark change records in the online log for replication. The Extractor reads only the records that are marked for replication from the online log.

  When Microsoft SQL Server produces backup logs and truncates the online log, it retains the records that are marked for replication in the online log. To mark these records as replicated, execute the SQL Server sp_repldone procedure. Thereafter, Microsoft SQL Server can delete the replicated records from the online log during the next backup operation.

  If you use the SQL Server Backup task to back up SQL Server logs, Data Replication by default executes the sp_repldone procedure to prevent unlimited growth of the log.

Note: When Change Data Capture is enabled, Microsoft SQL Server generates CDC capture tables. Data Replication does not use these SQL Server CDC tables for replication because it reads the transaction logs directly. Unless you use native Microsoft SQL Server replication, you can manually drop the SQL Server CDC capture tables.

Review the following information about enabling and disabling Change Data Capture for Microsoft SQL Server sources:

- To manage Change Data Capture for source databases and tables from the Data Replication Console, run the Server Manager on the Microsoft SQL Server system or on a remote system.
- If you have Microsoft SQL Server Enterprise Edition, the Server Manager enables Change Data Capture for the mapped source tables when you save the configuration from the Data Replication Console. Typically, you do not need to enable Change Data Capture manually from the Data Replication Console. However, you can view and edit CDC settings for mapped source tables in the configuration on the **Map Tables** tab. For more information, see "Managing Microsoft SQL Server Change Data Capture Settings" on page 344. You can also enable or disable SQL Server Change Data Capture by using SQL scripts. For more information, see Appendix G, "Sample Scripts for Enabling or Disabling SQL Server Change Data Capture" on page 456.
- If you have Microsoft SQL Server Standard Edition, you must enable Change Data Capture manually from the Data Replication Console. Enable Change Data Capture on the **Map Tables** tab for the mapped source tables before you create the replication configuration. Save the SQL script that contains the Change Data Capture statements and run the scripts manually on the server. For more information, see "Managing Microsoft SQL Server Change Data Capture Settings" on page 344.

  A Microsoft SQL Server Standard Edition instance must be restarted for Change Data Capture to be enabled. When enabling Change Data Capture, select the **Restart instance** option to have the Server Manager restart the instance. For the Server Manager to restart the instance, you must be connected to the Server Manager Main server as the idradmin user.

  Before you enable Change Data Capture for a Microsoft SQL Server Standard Edition instance, ensure that no dependent services, such as the Microsoft SQL Server Agent, are running on the source system. Active dependent services prevent the Server Manager from restarting the SQL Server instance and enabling Change Data Capture.
- If you have Microsoft SQL Server Enterprise Edition and select an SQL Server database in the **Source Schema** field on the **Map Tables** tab that includes tables for which Change Data Capture is not yet enabled, the Data Replication Console creates an auxiliary table in this database and enables Change Data Capture for the auxiliary table. When you save the configuration later, the Data Replication Console drops the auxiliary table and enables Change Data Capture for the mapped source tables. This behavior ensures that the Extractor can extract DDL and DML records starting from the LSN value that corresponds to the point at which Data Replication retrieved metadata from the source database. The metadata retrieval point corresponds to when you selected the source database on the **Map Tables** tab. The default name of the auxiliary table is IDR_DUMMY_1. To use another name for the auxiliary table, add the mssql_auxiliary_table_name_1 parameter in the *DataReplication_installation*/uiconf/default.cfg file. If the default.cfg file does not exist, use a text editor to create the file in the uiconf directory and then add the parameter in the file. For more information, see "Default.cfg File" on page 391.

  Note: The Extractor does not capture change data from the transactions that were open when the Data Replication Console enabled Change Data Capture for the auxiliary table.
- The Server Manager stores Change Data Capture settings for the mapped source tables in the Server Manager SQLite database, SM.db3. The Server Manager uses the database ID and table IDs to map source tables and their Change Data Capture settings. The Server Manager updates Change Data Capture settings for mapped source tables when you click the **Manage Database Supplemental Logging** icon button on the **Map Tables** tab. This opens the **Manage SQL Server Change Data Capture Settings** dialog box. From this dialog box, you can manage Change Data Capture settings for Microsoft SQL Server instances. Click **Refresh** to update the Change Data Capture settings for the selected schema. For more information, see "Managing Microsoft SQL Server Change Data Capture Settings" on page 344.

  When you create a Microsoft SQL Server database, SQL Server might assign a database ID to it that was previously used for a database that has been deleted. If you previously enabled Change Data Capture for the tables that were in the deleted database, you must update the Change Data Capture settings in the

Server Manager SQLite database before creating a replication configuration that maps the tables in the new database. If you do not update this information, the Server Manager might not enable Change Data Capture for the mapped source tables in the new database when you save the configuration in the Data Replication Console.

- For Microsoft SQL Server Enterprise Edition, SQL Server generates a CDC table for each table for which you enable SQL Server Change Data Capture. SQL Server uses the following pattern to name these CDC tables: `[cdc].owner_table_name_CT_object_ID`. SQL Server also generates two jobs named **[cdc].*database_name*_capture** and **[cdc].*database_name*_cleanup** for each database for which Change Data Capture is enabled. Data Replication does not use these SQL Server CDC tables and jobs for replication because it reads the transaction logs directly.
  By default, after the Server Manager enables Change Data Capture for source tables, it does not disable the SQL Server CDC jobs. If you run Data Replication and SQL Server Change Data Capture against the same database, you must disable Data Replication management of the secondary truncation checkpoint on the **Runtime Settings** tab > **General** view. For more information, see "Considerations for Managing the Secondary Truncation Checkpoint" on page 72. Also, ensure that the Extractor reads both the transaction logs and backup logs to prevent incomplete data capture.

  If SQL Server Change Data Capture or native replication is not active on the source database, ensure that Data Replication management of the secondary truncation checkpoint is enabled on the **Runtime Settings** tab > **General** view.

  If you want to override the default Server Manager behavior of handling CDC jobs, you can set the Server Manager **CdcJobsDisableMode** advanced property to 1 or 2 or select **Disable only if CDC is not enabled** or **Always disable** from the **SQL Server CDC Jobs** list in the **Microsoft SQL Server Instances** dialog box.
- If you drop a Microsoft SQL Server Enterprise Edition source table for which Change Data Capture was previously enabled, Microsoft SQL Server does not drop the corresponding CDC table. If you drop multiple source tables, drop the corresponding CDC tables manually to avoid unnecessary overhead on the source system.
- To enable Change Data Capture for source tables in a SQL Server Failover Cluster environment, ensure that the SQL Server Network Name matches the local definition of the name of the server on which the SQL Server source database runs. In the Failover Cluster Manager, you can determine the local definition of the server instance by issuing the following query: `SELECT @@SERVERNAME`. If the resulting local definition of the server instance does not match the SQL Server Network Name, change the name of the local server by using the sp_dropserver and sp_addserver stored procedures.

Related Topics:

- "Editing Microsoft SQL Server Instance Settings" on page 135
- "Managing Microsoft SQL Server Change Data Capture Settings" on page 344

Considerations for Managing the Secondary Truncation Checkpoint

After you enable Change Data Capture for at least one Microsoft SQL Server database, Data Replication and SQL Server backups of transaction logs use the secondary truncation checkpoint that is set for the database to identify the last replicated transaction in a transaction log. Data Replication and SQL Server back up only the log records that are marked as replicated.

Only one Data Replication or SQL Server task can manage the secondary truncation checkpoint. The Data Replication task can be an Extractor task or a Microsoft SQL Server Backup task. If multiple tasks try to manage the secondary truncation checkpoint, incomplete data capture might occur.

The Extractor runs the SQL Server sp_repldone procedure to update the secondary truncation checkpoint and mark extracted records as replicated.

Important: The sp_repldone procedure might take a long time to perform this processing if a large number of SQL operations occurred on the source since the last sp_repldone procedure run.

Disable secondary truncation checkpoint management by the Extractor in the following cases:

- You run the Extractor that reads transaction logs and SQL Server Change Data Capture (CDC) or native transactional replication against the same database. In this case, SQL Server must manage the secondary truncation checkpoint and produce backup logs.
- You run two Extractors that read transaction logs from the same database. Only one Extractor task can manage the secondary truncation checkpoint.
- You run the Extractor that reads transaction logs and a Data Replication Microsoft SQL Server Backup task for which the **Run the sp_repldone procedure** option is selected against the same database. Only one task can manage the secondary truncation checkpoint.

Important: If you disable the secondary truncation checkpoint management by the Extractor, ensure that the Extractor reads both the transaction and backup logs to prevent incomplete data capture.

To disable secondary truncation checkpoint management for the Extractor from the Data Replication Console, click the **Runtime Settings** tab > **General** view and select **Disable secondary truncation checkpoint**.

If none of the SQL Server and Data Replication tasks manage the secondary truncation checkpoint, log backups do not free data in the transaction log. This situation causes the transaction log to increase in size. To truncate transaction logs, perform one of the following actions:

- Enable secondary truncation checkpoint management for the Extractor and then run the Extractor.
- When you create the Microsoft SQL Server Backup task in the Data Replication Console, select the **Run the sp_repldone procedure** option.
- Mark the committed data in the transaction logs as "distributed" by running the sp_repldone procedure and then back up the transaction log. To perform these tasks, run the following commands:

  ```
  EXEC sp_repldone @xactid = NULL, @xact_segno = NULL, @numtrans = 0, @time = 0, @reset = 1
  BACKUP LOG database_name TO DISK = 'backup_log_file_name'
  ```

  Ensure that you back up the transaction log to a new backup file.

Supported DDL Operations for Microsoft SQL Server Sources

The Extractor for Microsoft SQL Server sources can process the following DDL operations:

- `ALTER TABLE table_name ADD column_name datatype [IDENTITY] [NULL|NOT NULL] [UNIQUE]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name PRIMARY KEY {CLUSTERED|NONCLUSTERED}`
- `ALTER TABLE table_name ALTER COLUMN column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name DROP COLUMN column_name`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name`
- `CREATE [UNIQUE] [CLUSTERED] INDEX index_name ON table_name (column_name)`
- `CREATE TABLE table_name (column_name datatype [IDENTITY] [NULL|NOT NULL] [UNIQUE])`
- `DROP INDEX index_name ON table_name`
- `DROP TABLE table_name`
- `SELECT * INTO new_table_name FROM existing_table_name`
- `TRUNCATE TABLE table_name`

Notes:

- Data Replication incorrectly replicates a SELECT * INTO operation if the source columns have LOB datatypes such as IMAGE, NTEXT, NVARCHAR(MAX), TEXT, VARBINARY(MAX), and VARCHAR(MAX).
- Data Replication can replicate DDL operations that include identity columns. When replicating these DDL operations to Oracle targets, which do not use identity columns, Data Replication replaces identity columns with standard table columns.
- Microsoft SQL Server allows TRUNCATE TABLE operations only on tables for which SQL Server Change Data Capture (CDC) is disabled. If you need to truncate a source table from which Data Replication extracts data, you must perform the following steps:
  1. Stop the Extractor task.
  2. Manually disable SQL Server CDC for the table that you want to truncate. Do not disable CDC at the database level.

     Important: While SQL Server CDC is disabled, Data Replication does not process DML changes that occur on the table.
  3. Perform the TRUNCATE TABLE operation.
  4. Enable SQL Server CDC again.
  5. Start the Extractor task.
- After a CREATE TABLE operation occurs on the source, you must manually enable Change Data Capture for this table to replicate change data to the target.

Related Topics:

- "Replication Considerations for Microsoft SQL Server Sources" on page 68
- "Editing Microsoft SQL Server Instance Settings" on page 135

MySQL Sources

Preparing MySQL Source Systems

After you install Informatica Data Replication on a Linux or Windows system that contains a MySQL source, you must complete several tasks to prepare the system for data replication.

1. For Windows systems, download and install the ODBC driver for MySQL. For Linux systems, use the DataDirect ODBC driver for MySQL that Data Replication provides in the *DataReplication_installation*/dd/lib subdirectory.
2. Use one of the following strategies to enable binary logging on the MySQL source database:
   - If you start the MySQL database server from the command line, enter the following command:

     ```
     mysqld --server-id[=server_id] --log-bin[=base_name] --binlog-format[=row]
     ```
   - If you start the MySQL database server as a service on Windows or Linux, you specify database configuration settings in an .ini or .cnf configuration file. The default file, my.ini, is located in the

MySQL installation directory. To enable binary logging, add the following lines to your MySQL configuration file:

```
[mysqld]
server-id=server_id
log-bin=base_name
binlog-format=row
```

Notes:

- For MySQL 5.7.3 and later, you must specify a server ID number greater than 0.
- Informatica recommends that you include the optional log-bin parameter to specify the base name for the sequence of binary log files. To create the binary log file names, MySQL adds a numeric suffix to the base name, which is incremented each time a new binary log is created. If you do not specify a base name, MySQL uses *host_name*-bin as the base name.
- Data Replication requires row-based binary logging. Verify that the binlog-format parameter is set to ROW before creating a Data Replication configuration that has a MySQL source. For MySQL 5.7 and later, the default binlog-format value is ROW.

3. Create a database user that Data Replication uses to connect to the source database. In the following example, the user name is datarep_user:

```
CREATE USER 'datarep_user'@'%' IDENTIFIED BY 'password';
```

4. Grant the following privileges to the user ID that Data Replication uses to connect to the MySQL source:
   - To enable the user to select data rows from tables in the database, grant the following privilege:

```
GRANT SELECT ON database_name.* TO 'datarep_user'@'%';
```

   - To enable the user to access local binary logs, grant the following privileges:

```
GRANT REPLICATION CLIENT ON *.* TO 'datarep_user'@'%';
GRANT SELECT, LOCK TABLES ON database_name.* TO 'datarep_user'@'%';
```

   - To enable the user to capture data from all databases, grant the following privileges:

```
GRANT SELECT, LOCK TABLES ON *.* TO 'datarep_user'@'%';
```

   - To enable the user to access remote binary logs, grant the following privilege:

```
GRANT REPLICATION SLAVE ON database_name.* TO 'datarep_user'@'%';
```

5. If the default.cfg file exists in the *DataReplication_installation*/uiconf/ directory, set the show_mysql_source_db_tab parameter to true in this file to enable support for MySQL sources.
6. To extract change data from a remote MySQL database server when the Extractor runs on a Red Hat Linux version 7.0 or later system, specify the lib64 directory before the $DBSYNC_HOME/support directory in the LD_LIBRARY_PATH environment variable.

   For example:

```
export LD_LIBRARY_PATH=/lib64:$DBSYNC_HOME/support:$LD_LIBRARY_PATH
```

   Important: An Extractor task that runs on Red Hat Linux 6.5 cannot extract change data from a remote MySQL database server.

Replication Considerations for MySQL Sources

Review the following replication considerations for MySQL sources:

- Data Replication supports MySQL source databases on Linux and Windows operating systems only.
- Data Replication supports MySQL source databases that use the InnoDB storage engine only.
- Data Replication supports InnoDB tablespace encryption of source tablespaces and Transparent Data Encryption (TDE) of InnoDB source tables.

- The following Data Replication features are not supported for MySQL sources:
  - Bidirectional replication
  - Importing or exporting Sync Point values
  - Managing open transactions from the Data Replication Console or Server Manger CLI
- Data Replication cannot replicate data from MySQL source tables that have columns with spatial datatypes such as GEOMETRY.
- Data Replication cannot replicate data from MySQL source columns that have the JSON datatype. However, Data Replication can replicate data from source tables that include columns with the JSON datatype, as long as these columns are not mapped in the replication configuration.
- If an Extractor task runs on Red Hat Linux 6.5, it cannot extract change data from a remote MySQL database server. However, if an Extractor task runs on Red Hat Linux 7.0 or later, it can extract data from a remote MySQL database server provided that you specify the lib64 directory before the $DBSYNC_HOME/support directory in the LD_LIBRARY_PATH environment variable.
- Data Replication captures change events from MySQL binary log files. Do not delete the binary log files until the Extractor has completed log file processing. When the Extractor shuts down, it reports the last log coordinates processed.
- After you run replication tasks for MySQL sources, Informatica recommends that you do not change the base name of the binary log files or change the name or location of the directory that contains the log files as specified on the **Extract Range** tab. If you must change the base log file name or change the binary log directory name or location, perform the following actions:
  1. Ensure that the change data in the binary log has been extracted, across all configurations with the MySQL source for which the binary log name will be changed.
  2. Stop all extraction tasks.
  3. To change the base name of the log files, rename the existing log files that were generated with the previous base name.
  4. To change the directory name or location, copy the existing binary log files to the new directory.
  5. From the **Extract Range** tab, edit the base name or directory in the **Base Bin Log File Name** field, for each configuration that includes the MySQL source to which the changes apply.
  6. Resume extraction processing.

Binary Log File

For MySQL sources, Data Replication uses the mysqlbinlog utility to read change events that are in binary format from binary log files. You specify the binary log base name and directory for a MySQL source connection on the **Extract Range** tab in the Data Replication Console.

MySQL binary log files contain events that describe changes, including table data changes and DDL changes such as CREATE TABLE. The term *binary log file* refers to an individual log file that contains database events and that has a unique name ending with a generated numeric suffix. The numeric suffix is incremented with each new log file. The term *binary log* refers to a series of binary log files plus the index file that contains the names of all used binary log files.

A binary log file name contains a base name and numeric suffix and has the following format: *base_name*_binlog.*numeric_suffix*. For example, mysql_5_7_binlog.000001.

When MySQL creates a new physical binary log file, it increments the numeric suffix in the log file name. MySQL creates a new log file under the following conditions:

- The MySQL server is restarted.

- The binary log file reaches the maximum size defined in the MySQL max_binlog_size variable.

If a MySQL server is restarted with a new log base name, a new series of binary logs starts. The first log file in the new series has a numeric suffix of 000001. Online binary log file name queries will not return information about the previous binary logs. However, the previous binary logs and index file remain available on the server disk.

Important: This scenario might cause data loss. To ensure data integrity, verify the current log base name when starting or restarting data capture.

Supported DDL Operations for MySQL Sources

The Extractor for MySQL sources can process the following DDL operations:

- `ALTER TABLE table_name ADD column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name PRIMARY KEY (column_name)`
- `ALTER TABLE table_name CHANGE old_column_name new_column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name DROP COLUMN column_name`
- `ALTER TABLE table_name DROP PRIMARY KEY`
- `ALTER TABLE table_name MODIFY column_name datatype [NULL|NOT NULL]`
- `CREATE [UNIQUE] INDEX index_name ON table_name (column_name)`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL] [PRIMARY KEY (column_name)])`
- `CREATE TABLE table_name AS SELECT statement`
- `DROP INDEX index_name ON table_name`
- `DROP TABLE table_name`

Oracle Sources

Preparing Oracle Source Systems

For Oracle and Amazon RDS for Oracle source systems, you must install Informatica Data Replication on a system that has read access to the Oracle redo log files. The Oracle database must be running in ARCHIVELOG mode with minimum global supplemental logging enabled.

Use one of the following strategies:

- To extract data from online redo logs, install and run the Extractor on the source Oracle server.
- To extract data from Oracle archive logs that are remote from the Extractor, either grant access to a network mapped share, such as NAS, NFS, or Samba, or FTP the archive log files to the machine where the Extractor is installed.

After you install Data Replication on an Oracle source system, you must perform the following tasks to prepare the system for data replication:

1. Verify that the Oracle Client is installed. The Data Replication Extractor and InitialSync components use the Oracle Client to access the Oracle logs.

   Important: The Oracle Client version must be equal to or later than the Oracle source database version.

2. Define the NLS_LANG parameter on the system where you run InitialSync. Set this parameter to match the Oracle source character set.

   Note: For configurations that have an Oracle source and Microsoft SQL Server target, always set the NLS_LANG parameter to UTF-8.

   If you use Informatica Fast Clone for initial materialization, see the *Fast Clone User Guide* for information about Fast Clone NLS_LANG configuration.

3. Verify that the ORACLE_HOME environment variable is defined and that the PATH environment variable includes the %*ORACLE_HOME*%\bin directory. Also, verify that an ORACLE_SID environment variable is defined for Oracle source instance.

4. Disable Automatic Diagnostic Repository (ADR) for Data Replication sessions that include the Oracle source so that InitialSync can generate a core file if a fatal error occurs.

   To disable ADR, add the following parameters to the sqlnet.ora configuration file:

   ```
   DIAG_ADR_ENABLED=OFF
   DIAG_DDE_ENABLED=FALSE
   DIAG_SIGHANDLER_ENABLED=FALSE
   ```

   By default, the sqlnet.ora configuration file is located in the ORACLE_HOME\network\admin directory.

   Tip: To disable ADR for Data Replication sessions only, copy the sqlnet.ora file to the *DataReplication_installation* directory and disable ADR in this sqlnet.ora file. Then use the TNS_ADMIN environment variable in an environment variables list to point to the *DataReplication_installation* directory.

5. For Amazon RDS for Oracle, create the ARCHIVELOG_DIR and ONLINELOG_DIR directories for Oracle redo logs. To create these directories, execute the following statements:

   ```
   exec rdsadmin.rdsadmin_master_util.create_archivelog_dir;
   exec rdsadmin.rdsadmin_master_util.create_onlinelog_dir;
   ```

   Tip: To prevent log discontinuity issues, define an appropriate retention policy for archived redo logs. Use the following procedure:

   ```
   exec rdsadmin.rdsadmin_util.set_configuration('archivelog retention
   days',number_of_days);
   ```

6. Enable Oracle ARCHIVELOG mode. See "Enabling ARCHIVELOG Mode" on page 78.

7. Enable Oracle minimum global supplemental logging. See "Enabling Minimal Global Supplemental Logging" on page 79.

8. Create a database user for connection to Oracle sources. See "Creating a Database User for Connecting to Oracle Sources" on page 79.

9. Configure Oracle user privileges. See "Configuring User Privileges for Oracle Sources" on page 79.

Enabling ARCHIVELOG Mode

Data Replication requires Oracle to be running in ARCHIVELOG mode.
By default, ARCHIVELOG mode is not enabled. To enable ARCHIVELOG mode, issue the following statements:

```
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
ALTER DATABASE ARCHIVELOG;
ALTER DATABASE OPEN;
SHUTDOWN IMMEDIATE;
STARTUP;
```

If you use the Oracle init.ora initialization parameter file, you must edit the appropriate parameters in this file to identify the archive log target and file name format. If you use a server parameter file (spfile), you must execute some ALTER SYSTEM SET SQL. For more information, see the Oracle Administrator Guide.

Enabling Minimal Global Supplemental Logging

Data Replication requires Oracle minimal global supplemental logging to be enabled.
To enable supplemental logging, execute one of the following SQL statements:

- For regular Oracle database installations:

```
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
```

- For Amazon RDS for Oracle:

```
exec rdsadmin.rdsadmin_util.alter_supplemental_logging('ADD');
```

Creating a Database User for Connecting to Oracle Sources

If you have Oracle sources, you must define a database user ID that has the authority to connect to Oracle to read redo logs.
On Linux or UNIX, you can use the “Oracle” user. Alternatively, you can create a database user, for example:

```
CREATE USER DATAREP_USER PROFILE "DEFAULT"
IDENTIFIED BY DATAREP_USER DEFAULT TABLESPACE "USERS"
TEMPORARY TABLESPACE "TEMP"
QUOTA UNLIMITED on "USERS" ACCOUNT UNLOCK;
```

Configuring User Privileges for Oracle Sources

Data Replication requires certain user privileges to read the Oracle online redo logs and archive logs and to perform other functions that involve database access. Grant the required user privileges to the user ID that Data Replication uses to connect to the Oracle source.

1. To retrieve the table structure of the source tables, grant the following privileges:

```
GRANT CREATE SESSION TO DATAREP_USER;
GRANT RESOURCE TO DATAREP_USER;
GRANT SELECT ANY TABLE TO DATAREP_USER;
GRANT SELECT ANY DICTIONARY TO DATAREP_USER;
GRANT ALTER SESSION TO DATAREP_USER;
GRANT EXECUTE ON DBMS_FLASHBACK TO DATAREP_USER;
```

   Note: You must grant EXECUTE on DBMS_FLASHBACK as the SYS user.

2. For Amazon RDS for Oracle, grant the following privileges:

```
GRANT EXECUTE ON rdsadmin.rds_file_util TO DATAREP_USER;
EXEC rdsadmin.rdsadmin_util.grant_sys_object('ARCHIVELOG_DIR', 'DATAREP_USER', 'READ');
EXEC rdsadmin.rdsadmin_util.grant_sys_object('ONLINELOG_DIR', 'DATAREP_USER', 'READ');
```

3. If you plan to replicate change data from encrypted Oracle 12*c* tables, grant SELECT on the SYS.ENC$ table.
   - For regular Oracle 12*c* database installations, execute the following SQL statement:

```
GRANT SELECT ON SYS.ENC$ TO DATAREP_USER;
```

   - For Amazon RDS for Oracle 12*c*, execute the following SQL statement:

```
EXEC rdsadmin.rdsadmin_util.grant_sys_object('ENC$', 'DATAREP_USER', 'SELECT');
```

4. To capture transactional data when running the InitialSync, grant the following privilege:

```
GRANT FLASHBACK ANY TABLE TO DATAREP_USER;
```

5. To add supplemental log groups for source tables from the Data Replication Console, grant the following privilege:

```
GRANT ALTER ANY TABLE TO DATAREP_USER;
```

6. If the Oracle database is in a RAC environment and your replication jobs do not read data from online redo logs, grant the following privilege:

```
GRANT ALTER SYSTEM TO DATAREP_USER;
```

This privilege enables the Extractor to force a log file switch on the RAC instances to get consistent log file data.

7. For InitialSync operations that use database links (dblinks) to synchronize Oracle sources and Oracle targets, grant the following privilege:

```
GRANT CREATE VIEW TO DATAREP_USER;
GRANT CREATE DATABASE LINK TO DATAREP_USER;
```

InitialSync creates a view for each source table or for each part of a source table that is processed on a separate thread. These views use the AS OF SCN clause to ensure data consistency. These views also include any WHERE clauses that are defined for the source columns to filter source data. After initial synchronization processing finishes, InitialSync drops these views.

Replication Considerations for Oracle Sources

Review the following information about Data Replication support for Oracle sources:

- Oracle must run in ARCHIVELOG mode and have minimal global supplemental logging enabled.
- Data Replication can extract change data from both Oracle online redo logs and archived logs in a Real Application Cluster (RAC) or non-RAC environment.
- Data Replication can extract change data from Oracle logs that are managed by Automatic Storage Management (ASM).
- Data Replication can extract change data from Oracle archived logs over the network.
- For Oracle sources that use ASM, Data Replication can read online redo logs over the network. The Extractor makes requests to ASM for data in the online redo logs. The Extractor can run on a system other than the Oracle source system.
- For Oracle sources without ASM, Data Replication can read online redo logs that are on a remote Network File System (NFS) in the following cases:
  - The Oracle database files reside on the NFS file system. Oracle accesses these files by means of the Oracle Direct NFS Client.
  - The Oracle database files reside on Network Attached Storage (NAS) devices that are certified by the Oracle Storage Compatibility Program (OSCP).
  - For NFS Version 3 and Version 4 file systems, the directory that contains the online redo logs is mounted with the following options: bg, hard, nointr, noac, and forcedirectio. If the forcedirectio option is not available, use the no_wdelay option or another mount option that provides direct I/O.
- Data Replication can extract change data from Amazon Relational Database Service (RDS) for Oracle deployments of Oracle databases in the cloud.
  To process redo logs from an Amazon RDS for Oracle instance, set the extract.oracle.rds_read_enable runtime parameter to 1.

  Important: For Amazon RDS for Oracle instances, the Extractor processes only the redo logs that are located in the following directories:

  - ARCHIVELOG_DIR for archived redo log files
  - ONLINELOG_DIR for online redo log files

- To extract change data from online redo logs remotely, export the directory that contains the online redo logs and mount the directory to the system where you run the Extractor. Use one of the following strategies to specify the location of the online redo logs for the Extractor:
  - On the system where you run the Extractor, mount the directory with the online redo logs to the same location that Oracle uses to access online redo logs. To get the location of the online redo logs for Oracle, execute the following statement:

    ```
    select * from v$logfile;
    ```
  - Use the ONLINE_FILES_PREFIX command line parameter for the Oracle Extractor to map the location of the online redo logs that Oracle uses to the location of online redo logs that the Extractor uses. Use this parameter if Oracle stores all of the online redo logs in one directory.
- Data Replication can extract changes from Oracle physical standby databases that are open in read-only mode and from Oracle logical standby databases.

  Important: If you use InitialSync to materialize an Oracle target based on a configuration that has an Oracle physical standby source, ensure that InitialSync does not use database links (dblinks) to unload the source data. Use the OCI library instead.
- Data Replication cannot extract change data from Oracle snapshot standby databases.
- Data Replication can extract change data from Oracle materialized views.
- Data Replication can extract change data from Oracle index-organized tables (IOTs), except partitioned IOTs defined with both the OVERFLOW and INCLUDING clauses.
- The Extractor cannot process Quick Multi Insert (QMI) operations for IOTs. Oracle produces QMI bulk-insert operations internally when you issue statements such as INSERT… SELECT and CREATE TABLE … AS SELECT. Therefore, the Extractor cannot extract INSERT … SELECT and CREATE TABLE … AS SELECT operations that use QMIs for IOTs.
- Data Replication can extract change data from Oracle compressed tables with the following limitations:
  - In Oracle 11*g* Release 1, Data Replication can only capture inserts, with or without the append hint, and direct path loads for tables that use FOR DIRECT_LOAD compression. Data Replication can only capture insert append operations and direct path loads for tables with OLTP compression in Oracle 11*g* Release 1. In Oracle 11*g* Release 2 and later, Data Replication can capture inserts, updates, and deletes, as normal, for tables with OLTP and FOR DIRECT_LOAD compression.
  - Data Replication does not support Oracle Exadata Hybrid Columnar Compression (EHCC).
- Data Replication supports Oracle Transparent Data Encryption (TDE) of source columns and tablespaces. Data Replication supports TDE with all of the Oracle encryption algorithms, including 3DES and AES with 128-bit, 192-bit, and 256-bit key lengths. Data Replication also supports TDE with MAC integrity checking and with SALT enhanced security.

  Before replicating change data from encrypted columns and tablespaces, you must import the required master keys from the Oracle wallet into the replication configuration. For more information, see "Enabling Replication from Oracle Tables and Tablespaces That Use Transparent Data Encryption" on page 257.

  When the Extractor loads the replication configuration, it decrypts the Oracle master keys that were imported into the configuration by using the internal master key. The Extractor uses the decrypted master keys to decrypt all of the TDE table keys that were also imported into the configuration. The Extractor then uses the TDE table keys to decrypt change data from encrypted columns.

  If the Extractor encounters an encrypted tablespace key in the redo log header, the Extractor decrypts this key by using the Oracle master key. The Extractor uses the TDE tablespace keys to decrypt redo records from encrypted tablespaces.

  The Extractor writes decrypted data to intermediate files. The Applier then reads the decrypted values and writes them to the target.

Important: The decryption of data from encrypted Oracle columns and tablespaces can significantly degrade Extractor performance.

If you change table encryption keys, Data Replication handles the changes automatically. You do not need to save the configuration file again. However, if you add a new encryption master key to the wallet, you must open the Oracle wallet and save the configuration again. Otherwise, the Extractor ends with an error because it cannot decrypt the encrypted column data.

- Data Replication can extract change data from Oracle online redo logs and archived logs on a raw device on Linux.
- If an Oracle RESETLOGS event occurs on a source database, Data Replication can continue extracting change data from the redo logs across the RESETLOGS boundary. A RESETLOGS event occurs in situations that require you to open the database with the RESETLOGS option, such as after a flashback database operation, incomplete point-in-time recovery, or point-in-time recovery with a backup control file. A RESETLOGS event can also occur transparently in a Data Guard environment with a physical standby database after a failover or after a switchover that is preceded by an incomplete recovery and followed by an ALTER DATABASE ACTIVATE PHYSICAL STANDBY DATABASE operation. A RESETLOGS operation archives the current online redo logs, resets the log sequence number to 1, creates a new database incarnation, creates a new timestamp and SCN for the online redo logs, and updates all the current data files with the new RESETLOGS SCN.
  When creating a processing queue for redo logs, the Extractor checks whether the database has undergone a RESETLOGS event. If a RESETLOGS event occurred, the Extractor determines whether it has processed redo records past the RESETLOGS boundary. If the Extractor has not passed the RESETLOGS boundary yet, the Extractor will process the remaining redo records for the old database incarnation and then, starting from the RESETLOGS boundary, will automatically start processing redo logs for the new database incarnation.

  If the Extractor has already passed the RESETLOGS boundary, the Extractor ends with an error. The error occurs because the Extractor already extracted change data that the Oracle source database rolled back after the RESETLOGS event. Consequently, data inconsistencies might occur.

  Tip: If the Extractor has already passed the RESETLOGS boundary, but the Applier has not applied the source data past this boundary yet, you can clean the replication configuration to resolve the problem. If the Applier has also passed the RESETLOGS boundary, you must run InitialSync to resynchronize the source and target.
- For configurations that have an Oracle source and target, Data Replication supports piecewise operations on LOB columns in SQL Apply mode.
  If the Oracle source uses inline LOBs, Data Replication supports piecewise operations in all apply modes for any supported target type. For targets other than Oracle, the Applier skips the following PL/SQL operations on inline and outline LOB columns:

  - APPEND, if the redo record does not contain the complete LOB data
  - ERASE
  - TRIM

  Tip: If the Oracle source uses piecewise operations that the Applier cannot process, set the apply.skip_oracle_piecewise_operations parameter to 1 or unmap the LOB columns that might be manipulated in pieces.
- Data Replication supports SecureFiles and BasicFiles storage of LOB columns.

  Restriction: Data Replication does not support SECUREFILE LOB columns with the following options:

  - DEDUPLICATE
  - FILESYSTEM_LIKE_LOGGING, if the size of LOB data exceeds 4 KB
  - NOLOGGING, if the size of LOB data exceeds 4 KB

- Data Replication does not support tables that were created with the ROWDEPENDENCIES setting.
- Data Replication supports virtual columns. The Data Replication Console shows virtual columns on the **Map Columns** tab.
- Data Replication does not support cluster, object, and XML tables.
- Data Replication does not support the replication of triggers and stored procedures.
- Oracle 11*g* Enterprise Edition introduced the Oracle Advanced Compression Option. For Oracle 11*g* Release 1 sources, Data Replication does not support the Advanced Compression Option. However, for later Oracle releases, Data Replication can extract change data from source tables that use the Advanced Compression Option.
- Data Replication cannot use a unique function-based index to uniquely identify rows when applying Updates and Deletes. If a source table does not have a primary key and includes only a unique function-based index, add a virtual index with the **Primary key** option that the Applier uses to uniquely identify target rows.
- AIX 6.1 introduced the open flag O_CIOR for file open system calls. Oracle 11*g* Release 2 (11.2.0.2) uses this flag to open redo log files. However, the O_CIOR flag cannot be used to open files on file systems that are mounted with the Concurrent IO (CIO) option. For Oracle 11*g* Release 2 sources on AIX 6.1 and later, ensure that the JFS2 file system that contains Oracle redo logs is mounted *without* the CIO option.
- Data Replication can extract change data from partitioned tables. However, if an ALTER TABLE partition operation occurs in the source, Data Replication might incorrectly replicate data to the target. The following DDL partition operations can affect data integrity on the target:
  - If a DROP PARTITION operation occurs on a source table, Data Replication does not delete the data that belonged to the dropped partition from the target.
  - If a SPLIT PARTITION operation occurs on a source table partition that contains data, Data Replication might load duplicate records to the target.
  - If a TRUNCATE PARTITION operation occurs on a source table, Data Replication does not delete the data that belonged to the truncated partition from the target.

  Tip: Perform partition-related DDL operations on mapped source tables that are defined with the NOLOGGING option. The Extractor then skips these operations.
- If you are replicating a partitioned source table and you perform an Oracle partition exchange operation between one of the partitions and a separate table, the following change data extraction limitations apply:
  - Data Replication does not replicate any DML changes that were written to the separate table before the partition exchange operation.
  - After the partition exchange, Data Replication replicates any new DML changes on a partition in the source table to the target.
  - If any rows are added to the partitioned table as a result of the partition exchange, Data Replication does not replicate these rows to the target because they are not logged in the redo log during the partition exchange. If you change or delete these rows and then use SQL Apply mode for data replication, apply errors occur because Data Replication detects a mismatch in the rows. However, you can configure Data Replication to ignore these errors on the **Runtime Settings** > **Error Handling** view. For more information, see "Configuring Applier Handling of Error Codes" on page 265.
  - If any rows are removed from the partitioned table as a result of the partition exchange, Data Replication does not remove these rows from the target because they are not logged in the redo log during the partition exchange.
  - If the partition exchange operation changes the content of the table and is not a simple reorganization or rebuild operation, Informatica recommends that you run InitialSync to resynchronize the source and target in the mapping.

- If an ALTER TABLE MOVE operation occurs on the source, Data Replication loads duplicate records to the target.

  Tip: Perform ALTER TABLE MOVE operations on mapped source tables that are defined with the NOLOGGING option. The Extractor then skips these operations.

- Data Replication supports the following Oracle 12*c* features:
  - Columns with extended datatypes of up to 32,767 bytes in size, including NVARCHAR2, RAW, and VARCHAR2 columns.
  - Tables for which Oracle In-Database Archiving is enabled. However, Data Replication does not replicate the ROW ARCHIVAL clause and any data or DDL operations for the hidden system column ORA_ARCHIVE_STATE.
  - Columns that are defined as invisible.
  - Numeric columns that are defined with the IDENTITY clause.
    If a source DDL statement contains the IDENTITY clause, the Applier executes the equivalent DDL statement on the target but without the IDENTITY clause.
  - Databases that use multitenant architecture. The source tables must reside in a single pluggable database (PDB) within a multitenant container database (CDB). In the Data Replication Console, you enter the name of the PDB on the **Source Database** tab in the **Instance** field.
- To replicate CREATE TABLE and CREATE INDEX operations from Oracle 12*c* sources, ensure that the _ORACLE_SCRIPT parameter is set to false for the Data Replication session that includes the Oracle source. To set this parameter to false, execute the following SQL statement:

```
ALTER SESSION SET _ORACLE_SCRIPT=false;
```

Related Topics:

- "Database Objects That Cannot Be Mapped" on page 219

Extractor Processing of Redo Logs from Oracle RAC Sources

Before you configure change data capture for an Oracle RAC source, review the following considerations that pertain to Extractor processing of redo logs in an Oracle RAC.

Order of Writing Extracted Records to the Intermediate Files

When extracting data from an Oracle RAC source, the Extractor preserves the order of the SCN values for the extracted records in the intermediate files.

Prior to writing a record from an online redo thread to an intermediate file, the Extractor compares the SCN value of this record with the SCN values of the latest records from other redo threads. If records for one or more online redo threads are not available to the Extractor, the Extractor cannot compare the SCN values and process available records. The Extractor then proceeds according to the mode in which it runs:

- In batch mode, the Extractor stops.
- In real-time mode, the Extractor tries to determine why the records are not available so that it can continue processing.

Configuring Archived Log Switches in an Oracle RAC Environment

For an Oracle RAC source, if you extract data only from archived redo logs, the Extractor requires an Oracle Call Interface (OCI) connection to the source. The Extractor uses the OCI connection to get information about

the redo threads and to perform an archived redo log switch before beginning change capture from these logs.

When you configure the connection to the Oracle RAC source from the **Source Database** tab, manually select the **Force log switch** option. This setting causes the Extractor to connect to the Oracle RAC source to get information about the redo threads and to perform an archived redo log switch. To perform an archived redo log switch, the Extractor executes the following SQL command:

```
ALTER SYSTEM ARCHIVE LOG CURRENT;
```

If you have an inactive Oracle instance that is not writing changes to online redo logs and is not archiving these logs, ensure that the Oracle RAC source is configured to have the Extractor perform forced log file switches. Otherwise, delays in extracting data from archived redo logs for other active Oracle instances might occur. The Extractor does not process the archived redo log records of active Oracle instances if the records have SCN values higher than the SCN values in the online logs of the inactive Oracle instance.

If the Extractor does not have an OCI connection to the Oracle RAC source, clear the **Force log switch** option. Then perform the following tasks:

- Switch archived redo logs prior to starting the Extractor. The log switch helps avoid delays in extracting change data.
- Ensure that the archived redo logs for all of the Oracle redo threads are available to the Extractor for the first Extractor run.
  The Extractor uses the archived redo logs to determine the redo threads to use for extracting data because it cannot get this information directly from the source. The Extractor saves the redo thread information to the configuration SQLite database. During subsequent runs, the Extractor retrieves information about the redo threads from the configuration SQLite database.

Supported DDL Operations for Oracle Sources

The Extractor for Oracle sources can process the following DDL operations:

- ALTER TABLE *table_name* ADD (*column_name datatype* [CONSTRAINT *constraint_name* {NOT NULL|UNIQUE}] [ENCRYPT|ENCRYPT USING|DECRYPT] [SALT|NO SALT])
- ALTER TABLE *table_name* ADD CONSTRAINT *constraint_name*
- ALTER TABLE *table_name* ADD PARTITION *partition_name* {VALUES|VALUES LESS THAN}(*value*) UPDATE INDEXES
- ALTER TABLE *table_name* DROP (*column_name*)
- ALTER TABLE *table_name* DROP CONSTRAINT *constraint_name*
- ALTER TABLE *table_name* DROP PARTITION *partition_name* UPDATE INDEXES
- ALTER TABLE *table_name* DROP SUBPARTITION *subpartition_name* UPDATE INDEXES
- ALTER TABLE *table_name* DROP {UNIQUE (*column_name*)|PRIMARY KEY}
- ALTER TABLE *table_name* MERGE PARTITIONS *partition_name*, *partition_name* INTO PARTITION *partition_name* UPDATE INDEXES
- ALTER TABLE *table_name* MERGE SUBPARTITIONS *subpartition_name*, *subpartition_name* INTO SUBPARTITION *subpartition_name* UPDATE INDEXES
- ALTER TABLE *table_name* MODIFY (*column_name datatype* [ENCRYPT|ENCRYPT USING|DECRYPT] [SALT|NO SALT])
- ALTER TABLE *table_name* MODIFY PARTITION *partition_name* ADD SUBPARTITION *subpartition_name* [VALUES (*value*)] UPDATE INDEXES

- ALTER TABLE *table_name* MOVE TABLESPACE *destination_tablespace_name* [PARALLEL *parallel_transaction_number*]
- ALTER TABLE *table_name* RENAME COLUMN *column_name* TO *new_column_name*
- ALTER TABLE *table_name* SPLIT PARTITION *partition_name* {AT|VALUES} (*value*) INTO (PARTITION *partition_name*, PARTITION *partition_name*) UPDATE INDEXES
- ALTER TABLE *table_name* SPLIT SUBPARTITION *partition_name* {AT|VALUES} (*value*) INTO (SUBPARTITION *subpartition_name*, SUBPARTITION *subpartition_name*) UPDATE INDEXES
- CREATE INDEX *schema_name.index_name* ON *schema_name.table_name*
- CREATE TABLE *table_name* (*column_name datatype* [CONSTRAINT *constraint_name* {NOT NULL| UNIQUE}] [ENCRYPT|ENCRYPT USING|DECRYPT] [SALT|NO SALT])
- CREATE TABLE *table_name* AS SELECT *statement*
- DROP INDEX *index_name*
- DROP TABLE *table_name*
- TRUNCATE TABLE *table_name*

Notes:

- To replicate CREATE TABLE and CREATE INDEX operations from Oracle 12*c* sources, ensure that the _ORACLE_SCRIPT parameter is set to false for the Data Replication session that includes the Oracle source. To set this parameter to false, execute the following SQL statement:

  ```
  ALTER SESSION SET _ORACLE_SCRIPT=false;
  ```

- For replication configurations that have an Oracle source and target, Data Replication can replicate index-organized tables (IOTs) to identical IOTs. However, if you have a target type other than Oracle, Data Replication replicates IOTs to regular target tables.
- Data Replication does not replicate CREATE TABLE ... AS SELECT statements that create IOTs.
- Data Replication does not replicate CREATE TABLE...AS SELECT * FROM operations if the replicated table contains LOB data.
- Data Replication does not replicate function-based indexes or function-based default values.
- Data Replication does not replicate indexes that are associated with a constraint. Data Replication replicates constraints without the associated index.
- Data Replication does not replicate the ENCRYPT keyword for Oracle source columns. On the target, Data Replication creates corresponding columns without the ENCRYPT keyword.
- Data Replication does not replicate temporary tables.
- Data Replication does not replicate CREATE INDEX operations that create an index in a schema that is different from the schema of the indexed table.
- Data Replication does not replicate DROP INDEX operations that drop an index in a schema that is different from the schema of the indexed table.
- Data Replication does not replicate the SECUREFILE option when processing ADD COLUMN operations for LOB columns.

CHAPTER 4

Targets - Preparation and Replication Considerations

This chapter includes the following topics:

Amazon Redshift Targets

Preparing Amazon Redshift Target Systems

After you install Data Replication on an Amazon Redshift target system, you must complete several tasks to prepare the system for data replication.

1. Install the ODBC driver.
   - On Windows, download and install the ODBC driver for PostgreSQL.
   - On Linux, download and install the Amazon Redshift ODBC driver.

2. To configure the Amazon Redshift ODBC driver on Linux, perform the following substeps:
   a. Verify the settings for the Amazon Redshift ODBC driver. In the [Redshift] section of the *DataReplication_installation*/dd/odbcinst.ini file, ensure that the path to the ODBC driver is correct. For example:

      ```
      [Redshift]
      Driver=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
      Setup=/opt/amazon/redshiftodbc/lib/64/libamazonredshiftodbc64.so
      ```

   b. Configure the Amazon Redshift ODBC driver to use UTF-8 instead of UTF-32. In the /opt/amazon/redshiftodbc/lib/64/amazon.redshiftodbc.ini file, set the DriverManagerEncoding parameter to UTF-8:

      ```
      [Driver]
      DriverManagerEncoding=UTF-8
      ```

3. Create an Amazon Simple Storage Service (Amazon S3) account.
4. Ensure that you have valid Amazon Web Services (AWS) credentials to access the Amazon S3 API.
5. Create an Amazon S3 bucket that will store the temporary files with the source data.

Replication Considerations for Amazon Redshift Targets

Review the following replication considerations for Amazon Redshift targets:

- Data Replication connects to and writes source data to the Amazon Simple Storage Service (Amazon S3). Amazon S3 is a storage service that can copy data from a source and simultaneously move the data to Amazon Redshift clusters. After the source data is in Amazon S3 storage, Data Replication issues a copy command that copies the data to an Amazon Redshift target table.
- Data Replication does not delete the temporary files that contain the source data from the Amazon S3 bucket if InitialSync or the Applier ends with an error. You can delete these files manually, or use Amazon S3 Object Lifecycle Management to automatically delete the temporary files from the bucket when the lifetime of the temporary files expires.
- Data Replication cannot replicate data from source columns that have binary datatypes to Amazon Redshift targets.
- Data Replication does not support replication from a single source to multiple Amazon Redshift targets. You cannot define multiple Amazon Redshift targets in a configuration. In the Data Replication Console, the **Routing** tab is unavailable when the target type is Amazon Redshift.

Supported DDL Operations for Amazon Redshift Targets

The Applier task can apply the following DDL operations to Amazon Redshift targets:

- `ALTER TABLE table_name ADD column_name datatype`
- `ALTER TABLE table_name ADD constraint_name PRIMARY KEY (column_name)`
- `ALTER TABLE table_name DROP COLUMN column_name CASCADE`
- `ALTER TABLE table_name DROP constraint_name CASCADE`
- `ALTER TABLE table_name RENAME COLUMN column_name TO new_name`
- `CREATE TABLE table_name (column_name datatype) [DISTSTYLE KEY DISTKEY (primary_key_column_name)|DISTSTYLE EVEN]`
- `DROP TABLE table_name`
- `TRUNCATE TABLE table_name`

Notes:

- To replicate CREATE TABLE and ADD COLUMN operations to Amazon Redshift targets in Audit Apply and Merge Apply modes, you must specify the new source table and column names in all lowercase. If you use uppercase, the Applier cannot replicate subsequent DML operations for the new table or column.
- Data Replication cannot replicate ADD COLUMN and CREATE TABLE operations that include columns with binary datatypes.
- Data Replication does not replicate ALTER COLUMN operations to Amazon Redshift targets because Amazon Redshift does not support these types of DDL operations. If you try to replicate ALTER COLUMN operations, the Applier ends with an error. Either do not enable replication of ALTER COLUMN operations or set the apply.skip_alter_column_failed_ddl runtime parameter to 1 to skip these errors.

Apache Kafka Targets

Preparing Apache Kafka Target Systems

To replicate change data to Apache Kafka targets, you must complete several prerequisite tasks to prepare the systems where the Applier and Apache Kafka target run.

1. Install the 64-bit Java Development Kit (JDK) 1.7 or 1.8 if you have not done so already.
2. Define the JAVA_HOME environment variable to point to the root Java installation directory.
3. Add a Java library to the system path.
   - On Windows, add the directory that contains the jvm.dll library to the PATH environment variable. For example, use the following command:

     ```
     PATH=%PATH%;%JAVA_HOME%\jre\bin\server
     ```

   - On Linux, add the directory that contains the libjvm.so library to the LD_LIBRARY_PATH path environment variable. For example, use the following command:

     ```
     LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/amd64/server
     ```

4. Verify that the Kafka producer.properties file for the Kafka installation is located in a directory on the target Server Manager where the Applier runs.

   Data Replication uses the producer.properties file to communicate with the Kafka target.
5. Ensure that the Kafka client library files match the version of Kafka to which Data Replication connects and are located in a directory on the machine where the Applier runs. Then, configure the files to work with Data Replication in one of the following ways:
   - Copy the Kafka client library files to the *DataReplication_installation*/lib/queueAdapterLib subdirectory.
   - After creating a configuration that uses the Kafka target, on the **Runtime Settings** tab > **Advanced Settings** view of the Data Replication Console, edit the apply.kafka.kafka_client_libraries_directory runtime parameter to specify the directory that contains the Kafka client library files on the machine where the Applier runs. For more information, see "Defining Apache Kafka Targets" on page 195 and Appendix C, "Data Replication Runtime Parameters" on page 396.

Replication Considerations for Apache Kafka Targets

Review the following limitations and replication considerations for Apache Kafka targets:

- Data Replication supports only Audit Apply mode for Kafka targets.
- InitialSync is not supported for Kafka targets.
- Data Replication does not replicate DDL changes, including TRUNCATE TABLE operations, to Kafka targets.
- Data Replication does not support SQL expressions for Kafka targets.
- Data Replication does not support the following replication topologies for Kafka targets: bidirectional replication, cascade replication, and replication from one source to multiple targets.
- Data Replication sends a single stream of messages that contain change data in commit order to Kafka targets. On the **Runtime Settings** tab > **General** view in the Data Replication Console, the only valid value for the **Applier threads** field is 1.
- The Kafka Applier uses the Java-based Queue Adapter component as an intermediary tool to process and format change data from a source database into Avro messages. Data Replication provides the required Queue Adapter library files in the *DataReplication_installation*/lib subdirectory.
- Data Replication supports only the following primitive Avro datatypes in Kafka messages:
  - NULL
  - BOOLEAN
  - INT
  - LONG
  - BYTES
  - STRING
- Data Replication does not support the use of the FLOAT or DOUBLE primitive Avro datatypes. The Applier converts all source data to characters in Kafka messages. Converting from a source FLOAT or DOUBLE datatype to characters and then to an Avro FLOAT or DOUBLE datatype might decrease data accuracy and precision.
- To minimize the risk of duplicated or missing messages on the Kafka target after a network outage occurs, the apply.kafka.kafka_producer_guarantee_delivery runtime parameter is enabled by default. Guaranteed delivery provides the highest level of data integrity but might degrade performance. To disable guaranteed delivery, set the apply.kafka.kafka_producer_guarantee_delivery runtime parameter to 0. With this setting, duplicate messages might occur on the Kafka target after you restart apply processing following an outage. The duplicate messages will have identical values for the OP_CMT_SCN (commit point) and OP_NUM_IN_TX (transaction ID) calculated columns in each message.
- For Apache Kafka targets, the Applier saves the sequence of the last change operation successfully sent to the target as a "checkpoint" to a checkpoint file, provided that you use the default guaranteed delivery mode. If you do not use guaranteed delivery, the Applier writes a checkpoint after each Commit operation. The checkpoint file must exist on the system where the Applier, the Queue Adapter, and a Server Manager instance (Main server or subserver) run. You can change the checkpoint file name or directory by editing the apply.kafka.kafka_checkpoint_file_directory and apply.kafka.kafka_checkpoint_file_name runtime parameters. By default, the checkpoint file name matches the configuration name.
- All Update operations that the Applier commits to Kafka targets generate UPDATE_EVENT messages in the Kafka topics, including Update operations that affect unmapped source columns or that result in no changes.

Cloudera and Hortonworks Targets

Preparing Cloudera and Hortonworks Target Systems

To replicate change data to Cloudera and Hortonworks targets on a Hadoop Distributed File System (HDFS), you must complete several prerequisite tasks to prepare the systems where the Applier and Data Replication Console run.

1. Install the 64-bit Java Development Kit (JDK) 1.7 or 1.8 if you have not done so already.

   Important: For Cloudera or Hortonworks targets that use Kerberos authentication, ensure that the JDK 1.7u65 or later is installed.
2. Define the JAVA_HOME environment variable to point to the root Java installation directory.
3. Add a Java library to the system path.
   - On Windows, add the directory that contains the jvm.dll library to the PATH environment variable. For example, use the following command:

     ```
     PATH=%PATH%;%JAVA_HOME%\jre\bin\server
     ```
   - On Linux and UNIX, add the directory that contains the libjvm.so library to the library path environment variable for your operating system. The library path environment variables are:
     - LD_LIBRARY_PATH for Linux systems
     - LD_LIBRARY_PATH_64 for Solaris systems
     - LIBPATH for AIX systems
     - SHLIB_PATH for HP-UX systems

     For example, use the following command:

     ```
     LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/amd64/server
     ```
4. On AIX, add the network API library, libnet.so, to the LIBPATH environment variable. Use the following command:

   ```
   LIBPATH=$LIBPATH:$JAVA_HOME/jre/lib/ppc_64
   ```
5. On Windows, install WinUtils by performing the following substeps:
   a. Download WinUtils from the following web site:

      https://github.com/steveloughran/winutils
   b. Extract the .zip file.
   c. Add the bin subdirectory that contains the WinUtils executable file and required .dll libraries to the HADOOP_HOME environment variable.
6. On Windows, for Cloudera or Hortonworks targets that use Kerberos authentication, define the DBSYNC_KERBEROS_CACHE_NAME environment variable. The environment variable points to the file that contains Kerberos credential cache.

   Tip: You can get the path to the Kerberos credential cache folder from the KRB5CCNAME environment variable.
7. Download the hadoop_libs.zip file that Data Replication provides and that contains the .jar files. Extract this zip file into the *DataReplication_installation* directory.
8. Verify that the *DataReplication_installation*/lib directory contains the hadoop subdirectory.

9. For Cloudera and Hortonworks targets, copy the following configuration files to the *DataReplication_installation*/lib/hadoop/*hadoop_distribution* directory:
   - hdfs-site.xml
   - core-site.xml
   - yarn-site.xml

     Note: The yarn-site.xml file is required only if the target uses HDFS high availability.

DB2 for Linux, UNIX, and Windows Targets

Preparing DB2 for Linux, UNIX, and Windows Target Systems

After you install Informatica Data Replication on a DB2 for Linux, UNIX, and Windows target system, you must complete several tasks to prepare the system for data replication.

1. Verify that either the DB2 Client 10.5 or DB2 Connect 10.5 is installed.

   Important: If the Applier task applies data to a remote DB2 database server, the DB2 Fix Pack version of the DB2 Client or DB2 Connect must be equal to or later than the DB2 Fix Pack version of the DB2 database.
2. To provide TCP/IP access to a remote DB2 server node that was not previously cataloged, issue the following DB2 catalog tcpip node command:

   ```
   db2 catalog tcpip node new_node_name remote target_server_ip server target_db2_port
   ```

   Note: The default DB2 port is 50000.

   To issue this command, you must have sysadm or sysctrl authority.
3. To store the location of a DB2 database that is on a remote node and that was not previously cataloged in the system database directory, issue the following DB2 catalog database command:

   ```
   db2 catalog database target_db2_database as new_db2_database_alias at node
   new_node_name authentication server
   ```

   The variable *new_db2_database_alias* name must be unique on each client machine.

   To issue this command, you must have sysadm or sysctrl authority.
4. Grant the following authorities and privileges to the user ID that Data Replication uses to connect to the DB2 target:
   a. Grant the sqladm, dataaccess, connect, createtab, and implicit_schema authorities. Use the following statements:

      ```
      db2 grant sqladm on database to user datarep_user
      db2 grant dataaccess on database to user datarep_user
      db2 grant connect on database to user datarep_user
      db2 grant createtab on database to user datarep_user
      db2 grant implicit_schema on database to user datarep_user
      ```

   b. Grant the privileges for creating, altering, and dropping objects within a target schema. Issue the following statement for each replicated schema:

      ```
      db2 grant alterin, dropin, createin on schema schema_name to user datarep_user
      ```

c. Grant the privilege required to create tables within a table space. Issue the following statement for each replicated table space:

```
db2 grant use of tablespace tablespace_name to user datarep_user
```

d. Grant the privilege required to apply change data to target tables. Issue the following statement for each replicated table:

```
db2 grant control on table table_name to user datarep_user
```

5. If you plan to perform bidirectional replication or cascade replication, perform one of the following steps:

- If you use DB2 9.7, run the following command to set the DB2_LOGGING_DETAIL registry variable for the target database instance to APPLINFO:

```
db2set -i <varname>instance_name</varname> DB2_LOGGING_DETAIL=APPLINFO
```

- If you use DB2 10.1 or later, run the following command to set the log_appl_info parameter for the target database to yes:

```
db2 update db cfg for database_name using log_appl_info yes
```

These commands support loopback avoidance by causing DB2 to write an additional informational log record with information about DB2 transactions. If you skip this step, the Server Manager cannot retrieve the following information for DB2 transactions: database transaction status, application ID, database user name, system user, and host. Also, the Extractor cannot distinguish the transactions that originally occurred on the database from the transactions that the Applier applied. As a result, the Extractor captures all of the change data.

Skip this step if you do not plan to use bidirectional or cascade replication.

Replication Considerations for DB2 for Linux, UNIX, and Windows Targets

Review the following replication considerations for DB2 targets:

- DB2 targets use a period (.) as the decimal separator to store float values. If you run an Applier or InitialSync task that uses the ODBC driver, ensure that the db2.odbc.enable_decimal_period runtime parameter is set to 1. If you set this parameter to 0, the ODBC driver uses the decimal separator that corresponds to the system locale settings. If the system locale settings use a decimal separator other than the period (.), the Applier and initialSync task cannot load float values to the target.
- If a DB2 target table does not contain a primary key or unique index, ensure that distribution by tables is set in one of the following ways:
  - On the **Map Columns** tab, select the **Distribute by tables** option.
  - On the **Map Tables** tab, click **Settings** to open the **Customize Settings for the Selected Tables** window. In the **Distribute By** column, select **Tables**.

  Note: Distribution by tables is the default option for DB2, Microsoft SQL Server, and Merge Apply targets. For Oracle targets, the default is distribution by rows.

Supported DDL Operations for DB2 for Linux, UNIX, and Windows Targets

The Applier task can apply the following DDL operations to DB2 for Linux, UNIX, and Windows targets:

- `ALTER TABLE table_name ADD column datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name {PRIMARY KEY|UNIQUE} (column_name)`
- `ALTER TABLE table_name ALTER COLUMN column_name SET DATA TYPE datatype`

- `ALTER TABLE table_name ALTER COLUMN column_name SET NOT NULL`
- `ALTER TABLE table_name ALTER column_name DROP NOT NULL`
- `ALTER TABLE table_name DROP column_name`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name`
- `ALTER TABLE table_name RENAME COLUMN column_name TO new_column_name`
- `CREATE [UNIQUE] INDEX index_name ON table_name`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL] [PRIMARY KEY] [UNIQUE])`
- `DROP INDEX index_name`
- `DROP TABLE table_name`
- `TRUNCATE TABLE table_name`

Notes:

- The Applier processes the CREATE INDEX statement only if this statement creates an index in the same schema as the parent table.

Greenplum Targets

Preparing Greenplum Target Systems

After you install Informatica Data Replication on a Greenplum target system, you must complete several tasks to prepare the system for data replication.

1. On Windows, download and install the ODBC driver for Greenplum. On Linux and UNIX, use the DataDirect ODBC driver for Greenplum that Data Replication provides in the *DataReplication_installation*/dd subdirectory.
2. Define the ODBCSYSINI environment variable in the user profile:

   ```
   ODBCSYSINI=$DBSYNC_HOME/dd
   ```

   Note: Define the DBSYNC_HOME environment variable to point to the Data Replication installation root directory if you have not done so.
3. On Linux and UNIX, add the ODBCSYSINI library in the library path environment variable for your operating system to define search paths for the dynamic linker.

   The library path environment variables are:

   - LD_LIBRARY_PATH for Linux systems
   - LD_LIBRARY_PATH_64 for Solaris systems
   - LIBPATH for AIX systems
   - SHLIB_PATH for HP-UX systems
4. Use the following statement to set and export the ODBCINST environment variable, which specifies the path and file name of the ODBC .ini file:

   ```
   EXPORT ODBCINST=$ODBCSYSINI/odbcinst.ini
   ```

   The odbcinst.ini system file contains information about the ODBC drivers that are available to all users.
5. Create a Greenplum database user for Data Replication. For example, use the name datarep_user.

6. Grant the following role and privileges to the datarep_user.

```
CREATE ROLE datarep_user LOGIN PASSWORD 'datarep_user';
GRANT ALL ON DATABASE <database_name> TO datarep_user;
GRANT CONNECT ON DATABASE <database_name> TO datarep_user;
```

7. Add the following lines for the IP address of Data Replication components to the PostgreSQL pg_hba.conf file in the Greenplum home directory:

```
HOST ALL datarep_user <IP_address_ DataRep_Console_system>/32 TRUST
HOST ALL datarep_user <IP_address_Applier_system>/32 TRUST
```

   The pg_hba.conf file is used for client authentication. For more information, see the PostgreSQL documentation.

Replication Considerations for Greenplum Targets

Review the following replication considerations for Greenplum targets:

- When replicating LOB data to Greenplum targets, the Applier and InitialSync truncate the LOB data to the size that is specified in the global.lob_truncation_size runtime parameter.
- By default, when InitialSync uses the native load utility to load source data to the target, the utility loads null values and empty strings as null values. Set the global.nz_pg_empty_strings_as_null runtime parameter to 0 to differentiate between empty strings and null values.

Supported DDL Operations for Greenplum Targets

The Applier task can apply the following DDL operations to Greenplum targets:

- `ALTER TABLE table_name ADD column_name datatype [NULL]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name PRIMARY KEY (column_name)`
- `ALTER TABLE table_name ALTER COLUMN column_name SET NOT NULL`
- `ALTER TABLE table_name ALTER COLUMN column_name TYPE datatype`
- `ALTER TABLE table_name DROP COLUMN column_name CASCADE`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name CASCADE`
- `ALTER TABLE table_name RENAME COLUMN old_column_name TO new_column_name` (Oracle sources only)
- `CREATE INDEX index_name ON table_name (column_name)`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL]) DISTRIBUTED RANDOMLY`
- `DROP INDEX index_name CASCADE`
- `DROP TABLE table_name`
- `TRUNCATE TABLE table_name`

Notes:

- For Oracle sources, Data Replication, by default, cannot replicate DROP CONSTRAINT operations that drop a primary key. When extracting these DDL operations, the Extractor sets the NULL option on the primary key columns. However, Greenplum does not support altering primary key columns.

  If you execute DROP CONSTRAINT operations that drop a primary key on the source, set the apply.skip_alter_column_failed_ddl runtime parameter to 1. This setting causes the Applier to not set the NULL option for the primary key columns on the target. You can then set the NULL option for these columns manually.

- To replicate CREATE TABLE, ADD COLUMN, and RENAME COLUMN operations to Greenplum targets in Audit Apply and Merge Apply modes, you must specify table and column names in all lowercase on the source. If you use uppercase, the Applier cannot replicate subsequent DML operations for the new table or column.

Microsoft SQL Server Targets

Preparing Microsoft SQL Server Target Systems

After you install Informatica Data Replication on a Microsoft SQL Server target system, you must complete several tasks to prepare the system for data replication.

1. Verify that the SQL Server database uses Mixed authentication mode to allow use of SQL Server Authentication.
2. Create a database user for Data Replication use. For example, use the user name “datarep_user.”
3. Set up an SQL Server login account for the database user that uses SQL Server Authentication.
4. Add the database user to the following server-level and database-level roles:
   - Server-level roles: dbcreator, public
   - Database-level role: db_owner. Add the Data Replication user to this role for each target database.
5. If you access SQL Server data from a Linux or UNIX system by using the Server Manager or another means, use the DataDirect ODBC driver that Data Replication provides in the *DataReplication_installation*/dd subdirectory.
6. Ensure that Microsoft SQL Server Native Client is installed on the system where InitialSync runs.

   InitialSync requires the Native Client to load data to Microsoft SQL Server targets by using the Bulk Copy Program (BCP).

Replication Considerations for Microsoft SQL Server Targets

Review the following replication considerations for Microsoft SQL Server targets:

- On Windows, by default, InitialSync uses the Bulk Copy Program (BCP) to load data to Microsoft SQL Server targets. If a mapped target table includes an unmapped NOT NULL column that has a DEFAULT clause, InitialSync transparently switches to the ODBC driver to circumvent BCP limitations in processing columns of this type.

  On Linux and UNIX, InitialSync uses the DataDirect ODBC driver to load data to Microsoft SQL Server targets.
- If InitialSync uses the Bulk Copy Program (BCP) to load data to Microsoft SQL Server targets, InitialSync requires audit log tables to process any SQL expressions that are defined for source tables in the configuration in SQL Apply mode.
- If you use the SQL Server Bulk Copy Program (BCP) to load data from Oracle source tables to the SQL Server target tables, InitialSync cannot use multiple subtask threads for any Oracle tables for which you defined virtual columns with Tcl scripts. In this situation, InitialSync ends with an error. To prevent this problem, set the initial.oracle.parallel_sample_percentage runtime parameter to the default value of 0. This setting causes InitalSync to process all table rows on a single thread.

- If you use multiple Applier threads and distribute changes across threads by rows, or if you use Applier subtask threads and the number of subtask threads is greater than the number of Applier threads, disable lock escalation for the mapped target tables and for the IDR_RECOVERY table. By disabling lock escalation, you minimize deadlocks and lock contention among the threads.

  To disable lock escalation for a table, issue the following DDL statement:

  ```
  ALTER TABLE table_owner.table_name SET (LOCK_ESCALATION=DISABLE)
  ```

  You can also minimize lock contention among the threads by creating a clustered index or primary key for each mapped target table.
- Data Replication cannot process numeric columns in a stored procedure that you created for a Custom resolution strategy. To define a conflict resolution rule that uses the Custom resolution strategy for a configuration with a Microsoft SQL Server target, manually replace numeric datatypes with character datatypes in the stored procedure definition. In the procedure body, convert the character data to a numeric datatype by using the appropriate database function.

Supported DDL Operations for Microsoft SQL Server Targets

The Applier task can apply the following DDL operations to Microsoft SQL Server targets:

- `ALTER TABLE table_name ADD column_name datatype [IDENTITY] [NULL|NOT NULL] [UNIQUE]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name PRIMARY KEY {CLUSTERED|NONCLUSTERED}`
- `ALTER TABLE table_name ALTER COLUMN column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name DROP COLUMN column_name`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name`
- `CREATE [UNIQUE] [CLUSTERED] INDEX index_name ON table_name (column_name)`
- `CREATE TABLE table_name (column_name datatype [IDENTITY] [NULL|NOT NULL] [UNIQUE])`
- `DROP INDEX index_name ON table_name`
- `DROP TABLE table_name`
- `SELECT * INTO new_table_name FROM existing_table_name`
- `TRUNCATE TABLE table_name`

Note: Data Replication replicates RENAME COLUMN operations from Oracle sources to Microsoft SQL Server targets as the following operation:

```
EXEC sp_rename 'table_name.[old_column_name]', 'new_column_name', 'COLUMN'
```

MySQL Targets

Preparing MySQL Target Systems

After you install Informatica Data Replication on a MySQL target system, you must complete several tasks to prepare the system for data replication.

1. For Windows systems, download and install the ODBC driver for MySQL. For Linux systems, use the DataDirect ODBC driver for MySQL that Data Replication provides in the *DataReplication_installation*/dd/lib subdirectory.

2. Define the ODBCSYSINI environment variable in the user profile:

```
ODBCSYSINI=$DBSYNC_HOME/dd
```

Note: Define the DBSYNC_HOME environment variable to point to the Data Replication installation root directory if you have not done so.

3. On Linux and UNIX, add the ODBCSYSINI library in the library path environment variable for your operating system to define search paths for the dynamic linker.

The library path environment variables are:

- LD_LIBRARY_PATH for Linux systems
- LD_LIBRARY_PATH_64 for Solaris systems
- LIBPATH for AIX systems
- SHLIB_PATH for HP-UX systems

4. Use the following statement to set and export the ODBCINST environment variable, which specifies the path and file name of the ODBC .ini file:

```
EXPORT ODBCINST=$ODBCSYSINI/odbcinst.ini
```

The odbcinst.ini system file contains information about the ODBC drivers that are available to all users.

5. Create a MySQL database user for Data Replication. For example, use the name datarep_user.

```
CREATE USER 'datarep_user'@'host_name' identified by 'password';
```

6. Grant the following role and privileges to the datarep_user.

```
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, LOCK TABLES
ON database_name.*
TO 'datarep_user'@'host_name';
```

Replication Considerations for MySQL Targets

Review the following replication considerations for MySQL targets:

- If the global.insert_buffer_size runtime parameter is set to the default value of 0 to dynamically determine the appropriate number of operations that can be buffered for the target type, the Applier buffers only 1 operation because of the MySQL ODBC driver restrictions. This setting causes the Applier to apply each SQL change operation from the source to the target instead of accumulating these changes in the internal buffer first. Enter a value other than the default value only at the request of Informatica Global Customer Support.
- Data Replication supports MySQL targets that use the InnoDB storage engine. If a MySQL target uses a non-transactional storage engine, recovery processing after an exception or failure during data replication might result in duplicate records or a row count mismatch error on the target.

Supported DDL Operations for MySQL Targets

The Applier task can apply the following DDL operations to MySQL targets:

- `ALTER TABLE table_name ADD column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name PRIMARY KEY (column_name)`
- `ALTER TABLE table_name CHANGE old_column_name new_column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name DROP COLUMN column_name`
- `ALTER TABLE table_name DROP PRIMARY KEY`
- `ALTER TABLE table_name MODIFY column_name datatype [NULL|NOT NULL]`

- `CREATE [UNIQUE] INDEX index_name ON table_name (column_name)`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL] [PRIMARY KEY (column_name)])`
- `DROP INDEX index_name ON table_name`
- `DROP TABLE table_name`

Note: Data Replication replicates TRUNCATE TABLE operations to MySQL targets as DELETE FROM operations.

Netezza Targets

Preparing Netezza Target Systems

After you install Informatica Data Replication on a Netezza target system, you must complete several tasks to prepare the system for data replication.

1. Download the Netezza ODBC driver from the Netezza Support website at http://www.netezza.com/support/index.aspx.
   a. Download the CLI package for your operating system from the Netezza FTP site. For example, download linux64cli.package.tar.
   b. Extract the contents of the tar file by using the tar command. For example:

      ```
      tar -xvf linux64cli.package.tar
      ```

      The following files are extracted:
      - npsclient.*version*.*patchlevel*.tar.Z
      - unpack
   c. Run the unpack shell script. Point to the location where the Netezza ODBC driver is installed. The expanded directories include a `lib64` directory and a `bin64` directory.
   d. For AIX, create a symbolic link in the lib64 directory:

      ```
      ln -s libnzodbc.a libnzodbc.so
      ```
2. To define the ODBC data source, define an ODBC driver named “NetezzaSQL” in the odbcinst.ini file.

   The following statements are required:

   ```
   [NetezzaSQL]
   Driver=$<Netezza_installation>/lib64/libnzodbc.so
   Threading=0
   ```

The following parameters are optional:

```
APILevel=1
CharacterTranslationOption=all
ConnectFunctions=YYN
DebugLogging=false
Description=Netezza ODBC driver
DriverODBCVer= 03.51
LogPath=/tmp
PreFetch=256
Setup=$<Netezza_installation>/lib64/libnzodbc.so
Socket=16384
UnicodeTranslationOption=utf8
```

If the odbcinst.ini file does not exist, you can create it in any directory.

3. Define the DD_INSTALLDIR, ODBCINST, and ODBCINI environment variables in the user profile:

```
DD_INSTALLDIR=$DBSYNC_HOME/dd
ODBCINST=$DD_INSTALLDIR/odbcinst.ini
ODBCINI=<user_home>/.odbc.ini
```

4. Add the ODBC environment variable values to the library path environment variable.

   The library path environment variable varies by operating system:

   - LD_LIBRARY_PATH on Linux
   - LD_LIBRARY_PATH_64 on Solaris
   - LIBPATH on AIX
   - SHLIB_PATH on HP-UX

   For example:

   library_path=$DBSYNC_HOME/support:*Netezza_driver_installation*/lib64:*library_path*

Replication Considerations for Netezza Targets

Review the following information about Data Replication support for Netezza targets:

- Because Netezza does not use binary datatypes, Data Replication does not support replication of these datatypes to Netezza targets. Before starting replication tasks for Netezza targets, ensure that the configuration does not map source columns that have binary datatypes to Netezza target columns.
- Netezza does not support Updates of distribution columns. Data Replication can process Updates to distribution columns if these columns are part of a logical primary key or if you used the DISTRIBUTE ON RANDOM clause when you created the Netezza target table. If you replicate Updates to Netezza distribution columns that are part of a logical primary key, the Applier processes all Updates in the current apply cycle as a pair of Delete and Insert operations.

  Note: If you do not specify the RANDOM option or the column names in the DISTRIBUTE ON clause for a Netezza target table, Netezza distributes data by the first column in the CREATE TABLE statement, by default. If you do not have a preferred distribution column, Informatica recommends that you distribute tables on primary key columns to support Updates of the distribution columns and to provide better performance in Merge Apply mode.
- When replicating a CREATE TABLE operation from a source to a Netezza target, Data Replication creates a table on the target and distributes data on the columns that are mapped to the primary key columns on the source. If the source table does not have a primary key definition, Data Replication uses the DISTRIBUTE ON RANDOM clause.

- When replicating LOB data to Netezza targets, the Applier and InitialSync truncate the LOB data to the size that is specified in the global.lob_truncation_size runtime parameter.
- On a Netezza target, the maximum number of tables that the Applier can load concurrently equals the number of Applier threads by default. You can specify the number of tables that the Applier can load concurrently in the apply.load_sessions_per_load_pass runtime parameter. To change this parameter value, Informatica recommends that you clear the **Create output as pipes instead of files** option on the **Runtime Settings** tab > **General** view. If this option is selected, each Applier thread can load only a single audit log table at a time.

  Important: Netezza supports a maximum of 31 concurrent Insert processes. To ensure that other applications can also load data to Netezza during data replication, the Applier limits the number of Netezza tables for concurrent load processing to a maximum of 20 tables.
- By default, when InitialSync uses the native load utility to load source data to the target, the utility loads null values and empty strings as null values. Set the global.nz_pg_empty_strings_as_null runtime parameter to 0 to differentiate between empty strings and null values.

Supported DDL Operations for Netezza Targets

The Applier task can apply the following DDL operations to Netezza targets:

- `ALTER TABLE table_name ADD column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name UNIQUE (column_name)`
- `ALTER TABLE table_name DROP COLUMN column_name CASCADE`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name CASCADE`
- `ALTER TABLE table_name MODIFY COLUMN column_name datatype`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL] [PRIMARY KEY (column_name)]) DISTRIBUTE ON {(column_name)|RANDOM}`
- `DROP TABLE table_name`

Notes:

- Data Replication does not support replication of DDL operations for source tables and columns that have lowercase names. The Applier ends abnormally when replicating subsequent DML changes to the corresponding tables and columns on a Netezza target.
- Data Replication does not replicate indexes that were created on a source database to Netezza targets. Netezza does not use indexes.
- Data Replication does not replicate ALTER COLUMN operations that add NULL and NOT NULL constraints for a source column to Netezza targets. Netezza does not support DDL operations of this type.
- Data Replication does not replicate ALTER COLUMN operations that modify a column datatype to Netezza targets. Netezza does not support DDL operations of this type.
- Data Replication does not replicate ALTER COLUMN operations that modify the length of columns that have a datatype other than CHAR, NCHAR, NVARCHAR, and VARCHAR to Netezza targets. Netezza does not support DDL operations of this type.
- Data Replication does not replicate ADD COLUMN operations that add a column that has a NOT NULL constraint but no default value from Oracle sources to Netezza targets. Netezza does not support this type of DDL operations.
- Data Replication replicates TRUNCATE TABLE operations from sources to Netezza targets as DELETE FROM operations.

- To replicate CREATE TABLE operation to Netezza targets, Data Replication creates a table on the target and distributes data on the columns that are mapped to the primary key columns on the source. If the source table does not have a primary key definition, Data Replication uses the DISTRIBUTE ON RANDOM clause.

Oracle Targets

Preparing Oracle Target Systems

After you install Informatica Data Replication on an Oracle target system, you must perform several tasks to prepare the system for data replication.

1. Verify that the Oracle Client is installed. The Data Replication Applier and InitialSync components use the Oracle Client to access the Oracle target tables.

   Important: The Oracle Client version must be equal to or later than the Oracle target database version.
2. To convert character sets between Oracle sources and Oracle targets, define the NLS_LANG parameter on the systems where you run the Applier and InitialSync tasks. Set this parameter to match the source character set.
3. Verify that the ORACLE_HOME environment variable is defined and that the PATH environment variable includes the ORACLE_HOME\bin directory. Also, verify that an ORACLE_SID environment variable is defined for Oracle source instance.
4. Disable Automatic Diagnostic Repository (ADR) for Data Replication sessions with the Oracle target so that the Applier or InitialSync can generate a core file if a fatal error occurs.

   To disable ADR, add the following parameters to the sqlnet.ora configuration file:

   ```
   DIAG_ADR_ENABLED=OFF
   DIAG_DDE_ENABLED=FALSE
   DIAG_SIGHANDLER_ENABLED=FALSE
   ```

   By default, the sqlnet.ora configuration file is located in the ORACLE_HOME\network\admin directory.

   Tip: To disable ADR for Data Replication sessions only, copy the sqlnet.ora file to the *DataReplication_installation* directory and disable ADR in this sqlnet.ora file. Then use the TNS_ADMIN environment variable in an environment variables list to point to the *DataReplication_installation* directory.
5. Create a database user for connecting to Oracle targets. See "Creating a Database User for Connecting to Oracle Targets " on page 102.
6. Configure database user privileges. See "Configuring Database User Privileges for Oracle Targets" on page 103.

Creating a Database User for Connecting to Oracle Targets

For targets, you must run Data Replication programs under a user ID that has the authority to write data to Oracle tables.
Create a Data Replication user, for example:

```
CREATE USER "DATAREP_USER" PROFILE "DEFAULT"
     IDENTIFIED BY "DATAREP_USER" DEFAULT TABLESPACE "USERS"
     TEMPORARY TABLESPACE "TEMP"
     QUOTA UNLIMITED on "USERS" ACCOUNT UNLOCK;
```

Configuring Database User Privileges for Oracle Targets

Informatica Data Replication requires certain user privileges to apply changes to Oracle target databases. Grant the following user privileges to the user ID that Data Replication uses to connect to the Oracle target:

1. To retrieve the table structure of the tables, grant the following privileges:

```
GRANT CONNECT                  TO DATAREP_USER;
GRANT RESOURCE                 TO DATAREP_USER;
GRANT SELECT ANY TABLE         TO DATAREP_USER;
GRANT SELECT ANY DICTIONARY    TO DATAREP_USER;
GRANT ALTER SESSION            TO DATAREP_USER;
```

2. To apply change data to target tables, grant the following privileges:

```
GRANT INSERT ANY TABLE         TO DATAREP_USER;
GRANT UPDATE ANY TABLE         TO DATAREP_USER;
GRANT DELETE ANY TABLE         TO DATAREP_USER;
```

3. For InitialSync operations that use database links (dblinks) to synchronize Oracle sources and Oracle targets, grant the following privilege:

```
GRANT CREATE DATABASE LINK     TO DATAREP_USER;
GRANT ALTER ANY TABLE          TO DATAREP_USER;
```

4. To generate target tables based on source tables, create recovery tables, and replicate DDL operations that create tables on the target, grant the following privilege:

```
GRANT CREATE TABLE                          TO DATAREP_USER;
```

5. For Audit Apply mode, grant the following privilege:

```
GRANT LOCK ANY TABLE      TO DATAREP_USER;
```

Replication Considerations for Oracle Targets

Review the following replication considerations for Oracle targets:

- If Oracle targets use Oracle 11.2.0.2 or later 11.2 patches, Data Replication can suppress Oracle database triggers so that they do not become active on target objects during replication. The Data Replication Oracle user must be granted special privileges. To grant these privileges, run the following Oracle stored procedure:

  - For Oracle 11.2.02 and later 11.2 patches, use dbms_goldengate_auth.grant_admin_privilege.

  For example, the following statement runs the stored procedure for Oracle 11.2.02 to grant the special privileges to the IDR_USER:

  ```
  exec dbms_goldengate_auth.grant_admin_privilege('IDR_USER');
  ```

  For Oracle 11*g* Release 1, Data Replication cannot suppress Oracle database triggers.

  To disable the suppression of triggers for Oracle targets, set the apply.disable_target_triggers parameter to 1 on the **Runtime Settings** tab > **Advanced Settings** view.

- If you set the apply.global_transaction_support runtime parameter to 0 on the **Runtime Settings** tab > **Advanced Settings** view, use one of the following methods to avoid deadlocks and lock contention among concurrent threads:

  - Configure the Oracle INITRANS parameter for the mapped target tables.

    - If you distribute changes across Applier threads by rows, set the Oracle INITRANS parameter for the mapped target tables to a value that is greater than or equal to the number of Applier threads plus two:

      ```
      INITRANS_value >= (number_of_threads + 2)
      ```

    - If you use Applier subtask threads, set the Oracle INITRANS parameter for the mapped target tables to a value that is greater than or equal to the number of Applier subtask threads plus two:

      ```
      INITRANS_value >= (number_of_subtask_threads + 2)
      ```

Important: You must reorganize the mapped target tables after changing the INITRANS value. If you do not reorganize the tables, the change to the INITRANS value does not apply to the existing table rows.

- Set distribution by tables for all of the tables and set the apply.subtasks_enabled runtime parameter to 0 to disable Applier subtask threads.

- For Oracle targets, you can use SQL Apply mode or Merge Apply mode. Informatica recommends that you use SQL Apply mode because it typically provides better performance. However, if the replicated tables contain bitmap indexes, Merge Apply mode is faster than SQL Apply.
- The Applier cannot replicate change data from any source type to Oracle LOB columns in Merge Apply mode. If the Applier attempts to do so, it fails.
- Oracle 11*g* Release 1 introduced system partitioning. For Oracle 11*g* Release 1 and later targets, Data Replication does not support replication of data to system partitioned tables.
- Data Replication uses the OCI direct path load interface to load data to audit log tables on Oracle targets. The OCI direct path load interface does not support loading data to tables with quoted names that include lowercase characters. If you use Merge Apply to load data to Oracle targets, ensure that the table names on the target do not include lowercase characters. If you use Audit Apply to load data to Oracle targets, set the apply.direct_load_for_audit_tables runtime parameter to 0 to use the OCI library instead of the OCI direct path load interface to load data to audit log tables that have quoted names.
- By default, the Applier and InitialSync replicate the source empty strings as null values to an Oracle target. If the source columns that include empty strings are mapped to the NOT NULL target columns, use the oracle.replace_empty_string_characters, oracle.replace_empty_string_enabled, and oracle.replace_empty_string_not_null_only runtime parameters to configure the Applier and InitialSync behavior and avoid the NOT NULL constraint violation. Set the oracle.replace_empty_string_enabled parameter to 1, and optionally edit the default values for the oracle.replace_empty_string_characters and oracle.replace_empty_string_not_null_only parameters. For more information, see Appendix C, “Data Replication Runtime Parameters” on page 396.

Supported DDL Operations for Oracle Targets

The Applier task can apply the following DDL operations to Oracle targets:

- `ALTER TABLE table_name ADD (column_name datatype [CONSTRAINT constraint_name {NOT NULL| UNIQUE}]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name`
- `ALTER TABLE table_name DROP (column_name)`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name`
- `ALTER TABLE table_name DROP {UNIQUE (column_name)|PRIMARY KEY}`
- `ALTER TABLE table_name MODIFY (column_name datatype`
- `ALTER TABLE table_name RENAME COLUMN column_name TO new_column_name`
- `CREATE INDEX index_name ON table_name`
- `CREATE TABLE table_name (column_name datatype [CONSTRAINT constraint_name {NOT NULL| UNIQUE}]`
- `CREATE TABLE table_name AS SELECT statement`
- `DROP INDEX index_name`
- `DROP TABLE table_name`
- `TRUNCATE TABLE table_name`

Notes:

- When replicating DDL changes from Microsoft SQL Server sources to Oracle targets, Data Replication does not replicate ALTER COLUMN operations that change a column datatype from VARCHAR(*n*) to VARCHAR(MAX). Data Replication maps the SQL Server VARCHAR(MAX) datatype to the Oracle CLOB datatype, but Oracle does not support an ALTER COLUMN operation that changes a VARCHAR column to a CLOB column.
- To replicate CREATE TABLE and ADD COLUMN operations to Oracle targets in Audit Apply and Merge Apply modes, you must specify table and column names in all uppercase on the source. If you use lowercase, the Applier cannot replicate subsequent DML operations for the new table or column.

PostgreSQL Targets

Preparing PostgreSQL Target Systems

After you install Informatica Data Replication on a PostgreSQL target system, you must complete several tasks to prepare the system for data replication.

1. On Windows, download and install the ODBC driver for PostgreSQL. On Linux and UNIX, use the DataDirect ODBC driver for PostgreSQL that Data Replication provides in the *DataReplication_installation*/dd subdirectory.
2. Define the ODBCSYSINI environment variable in the user profile:

   ```
   ODBCSYSINI=$DBSYNC_HOME/dd
   ```

   Note: Define the DBSYNC_HOME environment variable to point to the Data Replication installation root directory if you have not done so.
3. On Linux and UNIX, add the ODBCSYSINI library in the library path environment variable for your operating system to define search paths for the dynamic linker.

   The library path environment variables are:

   - LD_LIBRARY_PATH for Linux systems
   - LD_LIBRARY_PATH_64 for Solaris systems
   - LIBPATH for AIX systems
   - SHLIB_PATH for HP-UX systems
4. Use the following statement to set and export the ODBCINST environment variable, which specifies the path and file name of the ODBC .ini file:

   ```
   EXPORT ODBCINST=$ODBCSYSINI/odbcinst.ini
   ```

   The odbcinst.ini system file contains information about the ODBC drivers that are available to all users.
5. Create a PostgreSQL database user for Data Replication. For example, use the name datarep_user.
6. Grant the following role and privileges to the datarep_user.

   ```
   CREATE ROLE DATAREP USER LOGIN PASSWORD 'datarep_user';
   GRANT ALL ON DATABASE <database_name> TO datarep_user;
   GRANT CONNECT ON DATABASE <database_name> TO datarep_user;
   ```

7. Add the following lines for the IP address of Data Replication components to the PostgreSQL pg_hba.conf file:

```
HOST ALL datarep_user <IP_address_ DataRep_Console_system>/32 TRUST
HOST ALL datarep_user <IP_address_Applier_system>/32 TRUST
```

The pg_hba.conf file is used for client authentication. For more information, see the PostgreSQL documentation.

Replication Considerations for PostgreSQL Targets

Review the following replication consideration for PostgreSQL targets:

- If the global.insert_buffer_size runtime parameter is set to the default value of 0 to dynamically determine the appropriate number of operations that can be buffered for the target type, the Applier buffers only 1 operation because of the PostgreSQL ODBC driver restrictions. This setting causes the Applier to apply each SQL change operation from the source to the target instead of accumulating these changes in the internal buffer first. Enter a value other than the default value only at the request of Informatica Global Customer Support.
- The DataDirect ODBC driver for PostgreSQL does not support the BIGINT datatype. If you run the Applier on Linux or UNIX, ensure that the replication configuration does not map source columns to BIGINT columns.

  Note: *The DataReplication_installation*/uiconf/DataTypes.xml file includes rules that map source columns to BIGINT columns on the PostgreSQL target. If you run the Applier on Linux or UNIX, edit this file to map source columns to NUMERIC columns on the PostgreSQL targets before you generate the target schema.

Supported DDL Operations for PostgreSQL Targets

The Applier task can apply the following DDL operations to PostgreSQL targets:

- `ALTER TABLE table_name ADD column_name datatype`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name [PRIMARY KEY|UNIQUE] (column_name)`
- `ALTER TABLE table_name ALTER COLUMN column_name SET DATA TYPE datatype`
- `ALTER TABLE table_name ALTER COLUMN column_name SET NOT NULL`
- `ALTER TABLE table_name DROP COLUMN column_name CASCADE`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name CASCADE`
- `CREATE INDEX index_name ON table_name (column_name)`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL] [PRIMARY KEY (column_name))`
- `DROP INDEX index_name CASCADE`
- `DROP TABLE table_name`
- `TRUNCATE table_name`

Notes:

- To replicate CREATE TABLE, ADD COLUMN, and RENAME COLUMN operations to PostgreSQL targets in Audit Apply mode, you must specify table and column names in all lowercase on the source. If you use uppercase, the Applier cannot replicate subsequent DML operations for the new table or column.
- Data Replication does not enclose table names in quotation marks when replicating TRUNCATE TABLE operations. If the corresponding target tables have case-sensitive names, the Applier ends abnormally.

Teradata Targets

Preparing Teradata Target Systems

Complete this procedure to prepare Teradata targets for data replication use.

Data Replication requires the 64-bit Teradata ODBC driver and Teradata Parallel Transporter (TPT) libraries. Request the Teradata Tools and Utilities package that contains the Teradata ODBC driver and TPT libraries from the Teradata website. This package is not freely downloadable from Teradata.

Before contacting Teradata, gather the following information:

- The operating system and CPU Type of the Teradata target system where the libraries are going to be installed.
- The Teradata database version.

1. Ensure that the following Teradata client-side libraries are installed on the machine where the Data Replication Applier process will run:
   - Shared ICU Libraries
   - Teradata Call-Level Interface (CLIv2)
   - Teradata Data Connector API
   - Teradata GSS client package
   - Teradata BTEQ Utility
   - Teradata ODBC driver
   - TPT Base
   - TPT API
   - TPT Infrastructure
   - TPT Load Operator
   - TPT Stream Operator
   - TPT Update Operator

   Important: If TPT Steam Operator is not installed, you must set the teradata.stream_operator_enabled runtime parameter to 0.
2. Install the latest patches for the Teradata TPT, ODBC driver, and CLIv2 packages.
3. Update the PATH environment variable:
   - On Linux and UNIX systems, add the directory `/opt/teradata/client/version/tbuild/bin/` to the PATH environment variable.
   - On Windows systems, verify that the following directories are added to the PATH environment variable:

     ```
     C:\Program Files\Teradata\Client\version\bin\
     C:\Program Files (x86)\Teradata\Client\version\bin\
     ```

     Note: When you finish installing Teradata on a Windows system, the Teradata Tools and Utilities (TTU) 64-bit bin and 32-bit bin paths are the only paths added to the PATH environment variable. Teradata products do not depend on the PATH environment variable to find the TTU dependent libraries. However, third-party applications use the PATH environment variable to find Teradata shared libraries and executables.

4. On Linux and UNIX systems, set the following environment variables for the user in the .profile or .cshrc logon script:

```
ODBCINI=~/.odbc.ini
ODBCINST=/opt/teradata/client/version/odbc_64/odbcinst.ini
ODBCSYSINI=/opt/teradata/client/version/odbc_64
NLSPATH=/opt/teradata/client/version/odbc_64/msg/%N:/opt/teradata/client/version/
tbuild/msg64/%N
TPTAPI=/opt/teradata/client/version/tbuild
TDODBC=/opt/teradata/client/version/odbc_64
TDHOME=/opt/teradata/client
```

5. On Linux and UNIX systems, edit the LD_LIBRARY_PATH environment variable in the .profile or .cshrc logon script to include the following directories:

```
LD_LIBRARY_PATH=/opt/teradata/client/version/odbc_64/lib:/opt/teradata/client/
version/lib64:/opt/teradata/client/version/tbuild/lib64:$LD_LIBRARY_PATH
```

6. Set the Teradata CLIv2 environment variable THREADONOFF to 1 in the .profile or .cshrc logon script on the Linux or UNIX system where the target Server Manager instance runs. Use the following syntax:

```
THREADONOFF=1
```

Note: On Linux and UNIX systems, you must set the THREADONOFF environment variable to 1 to enable multithreaded support for the TPT API, which is used during data replication. If you use the default value of 0 for this environment variable, InitialSync and the Applier might end abnormally and generate a core file.

7. Create a database user for connecting to Teradata targets. See "Creating Database User Accounts for Connecting to Teradata Targets" on page 108.
8. Configure database user privileges. See "Configuring Database User Account Privileges for Teradata Targets" on page 108.

Creating Database User Accounts for Connecting to Teradata Targets

For targets, you must run Data Replication programs under a user account that has rights to read and update Teradata tables.

You can specify existing user accounts or create new user accounts for Data Replication, such as `DATAREP_USER`.

You can also create user accounts that are dedicated to specific Data Replication functions and manage the grants with security roles. For example, you could create a `DATAREP_USER_SYNCH` account for InitialSync tasks and a `DATAREP_USER_APPLY` account for Applier tasks.

Configuring Database User Account Privileges for Teradata Targets

Informatica Data Replication requires certain user privileges to read and apply changes to Teradata target databases. Target tables and the related log tables may be located in different databases. In that case, you must grant privileges on each database.

Grant the following privileges to the user accounts that Data Replication uses to connect to the Teradata target:

1. To retrieve the structure of the tables, grant the following privileges:

```
GRANT SELECT ON TARGET_DATABASE TO DATAREP_USER/ROLE;
GRANT SELECT ON LOG_DATABASE TO DATAREP_USER/ROLE;
```

2. To create new tables for replication or logging, grant the following privileges:

```
GRANT CREATE TABLE ON TARGET_DATABASE TO DATAREP_USER/ROLE;
GRANT CREATE TABLE ON LOG_DATABASE TO DATAREP_USER/ROLE;
```

Optionally, grant the following privileges if you want the user who created a table to also be able to drop the table:

```
GRANT DROP TABLE ON TARGET_DATABASE TO DATAREP_USER/ROLE;
GRANT DROP TABLE ON LOG_DATABASE TO DATAREP_USER/ROLE;
```

3. To insert or apply change data to target tables during InitialSync in Merge Apply mode, grant the following privileges:

```
GRANT INSERT ON TARGET_DATABASE TO DATAREP_USER/ROLE;
GRANT UPDATE ON TARGET_DATABASE TO DATAREP_USER;
GRANT DELETE ON TARGET_DATABASE TO DATAREP_USER;

GRANT INSERT ON LOG_DATABASE TO DATAREP_USER/ROLE;
GRANT UPDATE ON LOG_DATABASE TO DATAREP_USER/ROLE;
GRANT DELETE ON LOG_DATABASE TO DATAREP_USER/ROLE;

GRANT CREATE TABLE ON TARGET_DATABASE TO DATAREP_USER/ROLE;
GRANT DROP TABLE ON TARGET_DATABASE TO DATAREP_USER/ROLE;

GRANT CREATE TABLE ON LOG_DATABASE TO DATAREP_USER/ROLE;
GRANT DROP TABLE ON LOG_DATABASE TO DATAREP_USER/ROLE;

GRANT CREATE MACRO ON LOG_DATABASE TO DATAREP_USER/ROLE;
GRANT DROP MACRO ON LOG_DATABASE TO DATAREP_USER/ROLE;

GRANT CREATE VIEW ON TARGET_DATABASE TO DATAREP_USER/ROLE;
GRANT DROP VIEW ON TARGET_DATABASE TO DATAREP_USER/ROLE;
```

Note: The CREATE VIEW privilege is required only for handling long column names when log tables are created automatically. The corresponding DROP VIEW privilege is optional unless the user who created a view also needs privileges to drop the view. Data Replication does not perform any DROP VIEW actions.

4. To insert change data to log tables in Audit Apply mode, grant the following privileges:

```
GRANT INSERT ON LOG_DATABASE TO DATAREP_USER/ROLE;

GRANT CREATE TABLE ON LOG_DATABASE TO DATAREP_USER/ROLE;
GRANT DROP TABLE ON LOG_DATABASE TO DATAREP_USER/ROLE;
```

Note: CREATE TABLE and DROP TABLE privileges are required to automatically create the error, work, and trace tables that Teradata Parallel Transporter uses.

Replication Considerations for Teradata Targets

Review the following replication considerations for Teradata targets:

- When replicating LOB data to Teradata targets, the Applier and InitialSync truncate the LOB data to the size that is specified in the global.lob_truncation_size runtime parameter.
- To load data to Teradata tables, ensure that the row size does not exceed the TPT loader buffer of 64 KB. If you have source tables that include multiple LOB columns and expect the TPT loader buffer limit to be exceeded, set the global.lob_truncation_size runtime parameter to a smaller value. Also, edit the datatype conversion rules in the *DataReplication_installation*/uiconf/DataTypes.xml file to map the LOB columns to Teradata VARCHAR and VARBYTE columns of the appropriate length.
- The Applier might incorrectly apply Updates and Deletes to Teradata MULTISET tables that include multiple rows that contain the same primary key values from the source.
- When performing an initial synchronization of a source table with a Teradata target, if the source table contains duplicate rows and does not have a primary key or unique index, the corresponding Teradata target table must be defined as a NoPI (No Primary Index) table. NoPI tables are inherently multiset. If the target table is not defined as a NoPI table, InitialSync processing will fail with exit code 46.
- On a Teradata target, the maximum number of concurrent sessions for a TPT operator that an Applier thread can use for loading data to the target is determined by the value of the apply.teradata.td_max_sessions runtime parameter. This parameter value must be less than or equal to the number of Access Module Processors (AMPs) on the Teradata target database. The default value is 8.

Important: Teradata determines the maximum number of concurrent loads based on the settings in the Teradata options MaxLoadTasks and MaxLoadAWT.

- On a Teradata target, the maximum number of concurrent sessions for a TPT operator that an InitialSync thread can use for loading data to the target is determined by the value of the initialsync.teradata.td_max_sessions runtime parameter. This parameter value must be less than or equal to the number of AMPs on the Teradata target database. The default value is 8.
- If you use a version of the Teradata ODBC driver earlier than 15.00.00.04 and include special characters in the user name or password for connecting to the target, you must specify a custom connection string in the connection on the **Target Database** tab. In the custom connection string, enclose the user ID or password that contains the special characters in double quotation marks. Alternatively, use version 15.00.00.04 or a later version of the ODBC driver. These drivers can accept user IDs and passwords that include special characters, and you can connect to the Teradata target without specifying a custom connection string.

Supported DDL Operations for Teradata Targets

The Applier task can apply the following DDL operations to Teradata targets:

- `ALTER TABLE table_name ADD column_name datatype [NULL|NOT NULL]`
- `ALTER TABLE table_name ADD CONSTRAINT constraint_name UNIQUE (column_name)`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL]) PRIMARY INDEX (column_name)`
- `CREATE UNIQUE INDEX index_name (column_name) ON table_name`
- `DROP INDEX index_name ON table_name`
- `DROP TABLE table_name`

Notes:

- Data Replication does not replicate ALTER TABLE operations that modify a source column datatype to Teradata targets because Teradata does not support these types of DDL operations. Data Replication can only replicate operations that increase the length of character columns.
- Data Replication does not replicate DROP COLUMN operations for the source columns that are part of the primary key. For source tables that do not have a primary key, Data Replication does not replicate DROP COLUMN operations for the first column.
- Data Replication replicates TRUNCATE operations from sources to Teradata targets as DELETE ALL operations.

Vertica Targets

Preparing Vertica Target Systems

After you install Informatica Data Replication on a Vertica target system, you must complete several tasks to prepare the system for data replication.

1. Install the ODBC driver on the systems where you run the Applier and InitialSync.

- On Linux and UNIX, download the DataDirect ODBC driver for Vertica, libverticaodbc_datadirect.so. Then perform the following tasks:
  - Copy the driver file to the *DataReplication_installation*/dd/lib subdirectory.
  - Verify that the *DataReplication_installation*/dd/odbcinst.ini file provides the correct path to the driver file.
  - Run the server_manager.sh script to set all of the environment variables for the DataDirect ODBC driver.
- On Windows, download and install the Vertica ODBC driver.

2. On Linux and UNIX, define the VERTICAINI environment variable in the user profile on the systems where you run the Applier and InitialSync:

```
export VERTICAINI=$DBSYNC_HOME/dd/vertica.ini
```

Replication Considerations for Vertica Targets

Review the following replication consideration for Vertica targets:

- When replicating LOB data to Vertica targets, the Applier and InitialSync truncate the LOB data to the size that is specified in the global.lob_truncation_size runtime parameter.
- By default, when InitialSync uses the native load utility to load source data to the target, the utility loads null values and empty strings as null values. Set the global.nz_pg_empty_strings_as_null runtime parameter to 0 to differentiate between empty strings and null values.

Supported DDL Operations for Vertica Targets

The Applier task can apply the following DDL operations to Vertica targets:

- `ALTER TABLE table_name ADD CONSTRAINT constraint_name {UNIQUE|PRIMARY KEY} (column_name)`
- `ALTER TABLE table_name DROP CONSTRAINT constraint_name CASCADE`
- `ALTER TABLE table_name ADD column_name datatype`
- `ALTER TABLE table_name DROP COLUMN column_name CASCADE`
- `ALTER TABLE table_name ALTER COLUMN column_name SET NOT NULL`
- `ALTER TABLE table_name ALTER COLUMN column_name SET DATA TYPE datatype`
- `CREATE TABLE table_name (column_name datatype [NULL|NOT NULL] [PRIMARY KEY (column_name)]) [SEGMENTED BY HASH (column_name) ALL NODES]`
- `DROP TABLE table_name`
- `TRUNCATE TABLE table_name`

Notes:

- The Applier ends abnormally when applying a source ADD *column_name* operation that has the NULL, NOT NULL, or DEFAULT clause.
- Data Replication replicates ALTER COLUMN operations with the following limitations:
  - For target CHAR and VARCHAR columns that have no default values, Data Replication can change the column datatype and length.
  - For target BINARY and VARBINARY columns, Data Replication can change only the column length.

- For target INTEGER, INT, BIGINT, TINYINT, INT8, SMALLINT, and NUMERIC columns that have a precision of 0 and a scale less than or equal to 18, Data Replication can change any of the column properties.
- For target NUMERIC datatypes, Data Replication can change column scale values in the ranges (1-18), (19-37), and so on.

CHAPTER 5

Starting the Server Manager

This chapter includes the following topics:

Overview

You can run the Server Manager as a service or daemon or start the Server Manager executable manually each time you want to run it.

Note: The Server Manager must be running before you can connect to it in the Data Replication Console to perform data replication tasks.

Usually, the Server Manager runs as a service on Windows or as a daemon on Linux or UNIX. On Windows, running the Server Manager as a service ensures that the server is always running. You can also configure the Server Manager to start automatically after a system reboot. On Linux or UNIX, running the Server Manager as a daemon separates the Server Manager process from the terminal so that the process remains on the system even after the terminal is disconnected.

Installing the Server Manager as a Service on Windows

On Windows, optionally install the Server Manager as a service. Data Replication provides an installation script in its top-level installation directory that you can use.

1. Log in to the system as the Administrator user or log in with a user ID that belongs to the Administrator group.

2. At the command prompt, navigate to the top-level Data Replication installation directory that you specified in the DBSYNC_HOME environment variable.
3. Install the Server Manager as a service in one of the following ways:
   - Run the install_sm_service.cmd script.
   - Enter the following command:

     ```
     server_manager RUN_AS_SERVICE -i
     ```
4. If you install the Server Manager as a service for configurations that have a Microsoft SQL Server source, perform one of the following steps:
   - Add the built-in NT AUTHORITY\SYSTEM Windows account to the **sysadmin** fixed server role on the SQL Server source database.
   - Change the account under which the Server Manager service runs from the Local System account to a Windows account that is included in the list of SQL Server Logins. The account must be a member of the **sysadmin** fixed server role. Perform the following substeps:
     1. In Control Panel, click **System and Security** > **Administrative Tools** > **Services**.
     2. In the list of services, double-click the service for the Server Manager, which has the name **Informatica Data Replication *<version>***.
     3. In the **Service Properties** dialog box, click the **Log On** tab.
     4. Select **This account** and either enter or browse to the account that you want the service to use.
     5. Enter and confirm the password for this account.
     6. Click **OK**.

Starting the Server Manager as a Windows Service

Start the Server Manager as a service on Windows. Data Replication provides a script for starting the Server Manager as a service in the top-level installation directory.

1. At the command prompt, change directory (cd) to the *DataReplication_installation* directory using the following syntax: `cd %DBSYNC_HOME%`.
2. Run the start_sm_service.cmd script, or enter the following command at a command prompt:

   ```
   server_manager RUN_AS_SERVICE -s
   ```

   To run multiple instances of a Server Manager service on the same Windows machine, repeat step 1 for each top-level *DataReplication_installation* directory. Include the INSTANCE parameter in the command or script to specify a name for the service. For more information about this parameter, see "Command Line Parameters for the Server Manager" on page 448.

Starting the Server Manager as a Daemon on Linux or UNIX

Start the Server Manager as a daemon on Linux or UNIX.

1. At the command prompt, change directory (cd) to the *DataReplication_installation* directory using the following syntax: `cd $DBSYNC_HOME`.
2. Run the server_manager.sh script with the start parameter:

   ```
   ./server_manager.sh start
   ```

   Note: On Linux and UNIX, the Server Manager starts as a daemon by default if you run server_manager.sh without the start parameter. However, Informatica recommends that you include the start parameter to clearly indicate intent.

Manually Starting the Server Manager

If you do not run the Server Manager as a Windows service or a Linux or UNIX daemon, you can manually start it.

Use one of the following methods:

- On Windows, run server_manager.exe, which is located in the top-level installation directory.
- On Linux or UNIX, perform the following steps:
  1. Edit the server_manager.sh script to remove the RUN_AS_SERVICE parameter from the command that starts the Server Manager:

     ```
     if [ "$1" = "" ]
     then
       server_manager RUN_AS_SERVICE
     else
       server_manager RUN_AS_SERVICE $1 $2 $3 $4 $5 $6
     fi
     ```

  2. Save the script file.
  3. Run the script from the command line with the start parameter. Enter the following command:

     ```
     ./server_manager.sh start
     ```

Stopping a Server Manager Service or Daemon

If you are running the Server Manager as a service or a daemon and need to stop it, enter the appropriate command at the command prompt. Data Replication provides a script for stopping a Server Manager service in the top-level installation directory.

1. At the command prompt, change directory (cd) to the Data Replication installation directory that you specified in the DBSYNC_HOME environment variable.
   - On Windows, use the following syntax:

     ```
     cd %DBSYNC_HOME%
     ```

- On Linux and UNIX, use the following syntax:

  ```
  cd $DBSYNC_HOME
  ```

2. Stop the Server Manager in one of the following ways:
   - To stop a Server Manager service on Windows, run the stop_sm_service.cmd script, or enter the following command:

     ```
     server_manager.exe RUN_AS_SERVICE -k
     ```
   - To stop a Server Manager daemon on Linux or UNIX, enter the following command:

     ```
     ./server_manager.sh stop
     ```

     Then verify that the Server Manager daemon is not running. For example, use the following command:

     ```
     ps -eaf | grep server_manager
     ```

Uninstalling the Server Manager Service on Windows

If you need to uninstall a Server Manager service on Windows, enter the appropriate command at the command prompt.

1. Open the command prompt and then change directory (cd) to the top-level Data Replication installation directory that you specified in the DBSYNC_HOME environment variable. Use the following syntax:

   ```
   cd %DBSYNC_HOME%
   ```
2. Run the uninstall_sm_service.cmd script, or enter the following command:

   ```
   server_manager.exe RUN_AS_SERVICE -r
   ```

CHAPTER 6

Getting Started with the Data Replication Console

This chapter includes the following topics:

Data Replication Console Interface

The Data Replication Console is a graphical user interface. From the Data Replication Console, you can create replication configurations and schedules, start and stop replication schedules and tasks, and monitor replication jobs. Informatica recommends that you use the Console for creating configurations because it guides you through the configuration creation process and validates some entries.

You can run the Data Replication Console standalone on any Windows computer system as long as the Console has JDBC connectivity to the source or target system.

Note: To update the source and target metadata for existing configurations that are saved as XML files, you can use the Replication Configuration Command Line Interface (CLI) instead of the Data Replication Console. This CLI runs on a Linux, UNIX, or Window machine. For more information, see Appendix E, “Updating Configurations in the Replication Configuration CLI” on page 451.

Data Replication Console Tabs

The Data Replication Console main window contains multiple tabs at the top. On these tabs, you enter information to define a configuration for a data replication job or InitialSync task.

Some tabs have cross-tab views. Click a view to change the fields and options that are displayed on the tab. The views group fields and options by sub-function.

The following tabs appear in the main window:

Server Manager tab

From the many tab views, you can connect to a Server Manager Main server and configure replication tasks, source and target connections, schedules, and environment variables. Also, you can view replication statistics for performance monitoring.

Source Database tab

On this tab, enter connection information for a source database. The fields vary slightly based on the database type. For example, to connect to an Oracle database, the Console requires an Oracle instance name. However, to connect to DB2 for Linux, UNIX, and Windows, the Console requires a DB2 database name.

Target Database tab

On this tab, enter connection information for a target.

Map Tables tab

On this tab, map source tables to target tables. You can also customize some capture and apply settings, configure re-synchronization of selected target tables, and specify the SCNs or LSNs from which to start Extractor and Applier processing.

Map Columns tab

On this tab, edit mappings of source and target columns.

Routing tab

On this tab, you can specify additional target servers for a replication job that has multiple targets.

Extract Range tab

On this tab, specify the location of the database log files from which the Extractor extracts data.

Runtime Settings tab

On the many views of this tab, enter runtime settings for a replication job, including advanced runtime parameters.

Logging tab

On this tab, customize the types of messages that Data Replication adds to task execution logs that are located in the *DataReplication_installation*/logs directory.

Toolbar Icon Buttons

Instead of choosing commands from the menu bar or right-clicking an item and selecting a command, you can click icon buttons on the Console toolbars to quickly initiate common actions.

Main Toolbar Buttons

The following icon buttons appear on the main toolbar:

Button	Description
	Create a configuration file.
	Open an existing configuration file.
	Save a configuration file remotely.
	Run an InitialSync task.
	Run an Extractor task.

Button	Description
	Run an Applier task.
	Generate schema-definition scripts (DDL) for a target.
	Display latency, Extractor, and intermediate files statistics for the open configuration.
	Enable Edit mode for the configuration that is currently open.
	Enable Read mode for the configuration that is currently open.

Replication Configurations Toolbar Buttons

The following icon buttons appear on the Server Manager tab > **Configs** view:

Button	Description
	Create a new replication configuration.
	Delete the selected replication configuration.
	Edit properties for the selected replication configuration, including the owner, connections, Server Manager instances, and runtime environments.
	In the **Editing** dialog box that displays when you click the **Properties** icon button, change the source or target Server Manager instance for configuration connections.
	Import a replication configuration.
	Export a replication configuration for saving as an XML file and a set of DB files.
	Clean the selected replication configuration.
	Display the **Deploy a Configuration** wizard for deploying the selected replication configuration.
	Display latency, Extractor, and intermediate files statistics for the selected replication configuration.
	Display the revisions for a selected replication configuration.
	Display a list of processed database log files for the selected replication configuration.
	Display a list of intermediate files for the selected replication configuration.
	Generate a reverse configuration of the selected replication configuration for bidirectional replication.
	Show the open transactions for the selected replication configuration.

Button	Description
	Check the connection from the Server Manager Main server to any subservers and from subservers back to the Server Manager Main server.
	Edit the default Applier, Extractor, and InitialSync tasks for a configuration.

Server Managers Toolbar Buttons

The following icon buttons appear on the **Server Manager** tab > **Servers** view:

Button	Description
	Add a new Server Manager subserver instance.
	Edit connection details for the selected Server Manager instance.
	Delete the selected Server Manager instance.
	Display the message log for the selected Server Manager instance.
	For Microsoft SQL Server source instances on the system where the selected Server Manager runs, manage connection details and SQL Server CDC jobs.
	Display details about the system environment variables on the system where the selected Server Manager instance runs.
	Test the connection between the selected Server Manager instance and the Data Replication Console.
	Check the connection from the Server Manager Main server to any subservers and from subservers back to the Server Manager Main server.
	Display properties for the selected Server Manager system, including the available disk space. Enables you to edit properties such as file paths, Server Manager advanced settings, message log settings, and parameters for latency statistics.
	Display a list of available servers.

Schedules Toolbar Buttons

The **Server Manager** tab > **Schedules** view includes two toolbars.

Schedules toolbar:

Button	Description
	Create a new schedule.
	Edit details for the selected schedule.
	Delete the selected schedule.

Button	Description
	Start the Schedule Wizard for creating a new schedule.
	Start the selected schedule.
	Stop the selected schedule.
	Force the selected task to end immediately when you cannot wait for the task to complete processing.
	Display latency, Extractor, and intermediate files statistics for the replication configuration that is associated with the selected schedule.
	Display a list of intermediate files for the replication configuration that is associated with the selected schedule.
	Refresh the status of the selected schedule.

Processes toolbar:

Button	Description
	Display detailed information about a selected task, including the run ID, task name, start and end time, database connection, server, type of task, command syntax, directory, success or failure status, and exit code.
	Display the task output log file names and sizes.
	Display any processes that are blocking the selected task.
	Start the selected task.
	Display the process logs for the selected task.
	Stop the selected task.
	Force the selected task to end immediately when you cannot wait for the task to complete processing.

Tasks and Task Dependencies Toolbar Buttons

The following icon buttons appear on the **Server Manager** tab > **Tasks** view:

Button	Description
	Create a new task.
	Edit the details and task dependencies for the selected task.

Button	Description
	Delete the selected task.
	Select the tasks that block the selected task.

Connections Toolbar Buttons

The following icon buttons appear on the **Server Manager** tab > **Connections** view:

Button	Description
	Create a new connection.
	Edit the connection details for the selected connection.
	Delete the selected connection.
	Test the connection between the source or target database specified in the selected connection and the Data Replication Console.

Environment Lists Toolbar Buttons

The following icon buttons appear on the **Server Manager** tab > **Environment** view:

Button	Description
	Create a new environment variable.
	Edit the selected environment variable.
	Delete the selected environment variable.

Event Viewer > Logs of Schedules Toolbar Buttons

The following icon buttons appear on the **Server Manager** tab > **Event Viewer** view > **Logs of Schedules** subtab:

Button	Description
	Display detailed information about a selected task, including the run ID, task name, start and end time, database connection, server, type of task, command syntax, directory, success or failure status, and exit code.
	Display the task output log file names and sizes.
	Display any processes that are blocking the selected task.

Button	Description
	Display the process logs for the selected task.
	Stop the selected task.
	Force the selected task to end immediately when you cannot wait for the task to complete processing.

Map Tables Tab Toolbar Buttons

The following icon buttons appear on the **Map Tables** tab:

Button	Description
	Unmap a mapped source table.
	Edit the indexes that are defined for the selected mapped source table.
	Display, add, edit, delete, or reuse virtual columns for the selected mapped source table.
	Add or edit an SQL or Tcl expression for the selected source table.
	Define conflict resolution rules for the selected target table to handle conflicts that might occur during apply processing.
	Manage the database supplemental logging settings for DB2 for Linux, UNIX, and Windows, Microsoft SQL Server, and Oracle sources.

Map Columns Tab Toolbar Buttons

The following icon buttons appear on the **Map Columns** tab:

Button	Description
	Add or edit a WHERE clause to filter data from redo logs for a selected source column.
	Delete a virtual column.
	Add or edit an SQL or Tcl expression for a virtual column.
	Add a virtual column to the source table.

Routing Tab Toolbar Buttons

The following icon buttons appear on the **Routing** tab and its subtabs:

Top toolbar:

Button	Description
	Add a new target server instance and a connection for the target.
	Edit the connection for the selected target server instance.
	Delete the selected target server instance.

Filters subtab > toolbar for groups:

Button	Description
	Add a new filter group.
	Edit the selected filter group.
	Delete the selected filter group.

Filters subtab > toolbar for columns:

Button	Description
	Add a new filter condition for the group filter.
	Edit the selected filter condition.
	Delete the selected filter condition.

Schema Name subtab toolbar:

Button	Description
	Rename the target schema name.
	Edit the selected target schema name.
	Delete the selected target schema name.

Data Replication Console Log File

The Data Replication Console writes information about critical errors and exceptions to the *DataReplication_installation*/logs/se.log file. Use this file to diagnose any Console problems that might occur.

The first time you run the Data Replication Console, Data Replication creates the se.log file on the local file system. For each subsequent run of the Data Replication Console, Data Replication purges the log file to prepare for new messages for the current session.

Starting the Data Replication Console

Run the Data Replication Console either on a Windows source or target system or on a standalone Windows machine.

Before you start the Data Replication Console the first time, verify that the prerequisites are met.

To start the Data Replication Console on a Windows system, run idr_console.cmd, which is located in the *DataReplication_installation* directory.

CHAPTER 7

Defining and Managing Server Manager Main Servers and Subservers

This chapter includes the following topics:

Server Manager Main Server and Subservers

The Server Manager enables data movement from a source database to a target database over a LAN or WAN. The Server Manager acts as the main gateway for configuration, management, and monitoring of replication jobs.

You work with a Server Manager instance from the **Server Manager** tab of the Data Replication Console.

The master Server Manager server is called the *Main server*. Typically, you run the Main server on the system where the source database runs. You can run the Main server on the system where the target database runs or on another machine. You can optionally define several *subservers* for a Main server. Typically, subservers are located on remote machines where the Extractor or Applier run. The subservers run the replication tasks and transmit the intermediate files.

Note: After you connect to the Main server, you cannot later use it as a subserver. After you add a subserver, you cannot later use it as a Main server.

The Main server and the subservers that are associated with it comprise a *replication server group*. Each server can belong only to one replication server group. The Main server and its subservers store information

about the servers in their replication server group in the *DataReplication_installation*/SM.db3 SQLite file. The Server Manager identifies the servers by using a unique server key. A server accepts messages only from the other servers in the replication server group.

The Server Manager Main server and each subserver also store information about the role of the server in the replication group in the *DataReplication_installation*/SM.db3 SQLite file.

You can connect to the Server Manager Main server from the Data Replication Console or the Server Manager Command Line Interface (CLI). For more information about the Server Manager CLI, see the *Informatica Data Replication Command Line Interface for the Server Manager*.

After you connect to the Main server, you can perform the following server management tasks:

- Deploy configuration files
- Schedule replication jobs
- Monitor replication jobs
- Edit server properties

Defining the Main Server and Its Subservers

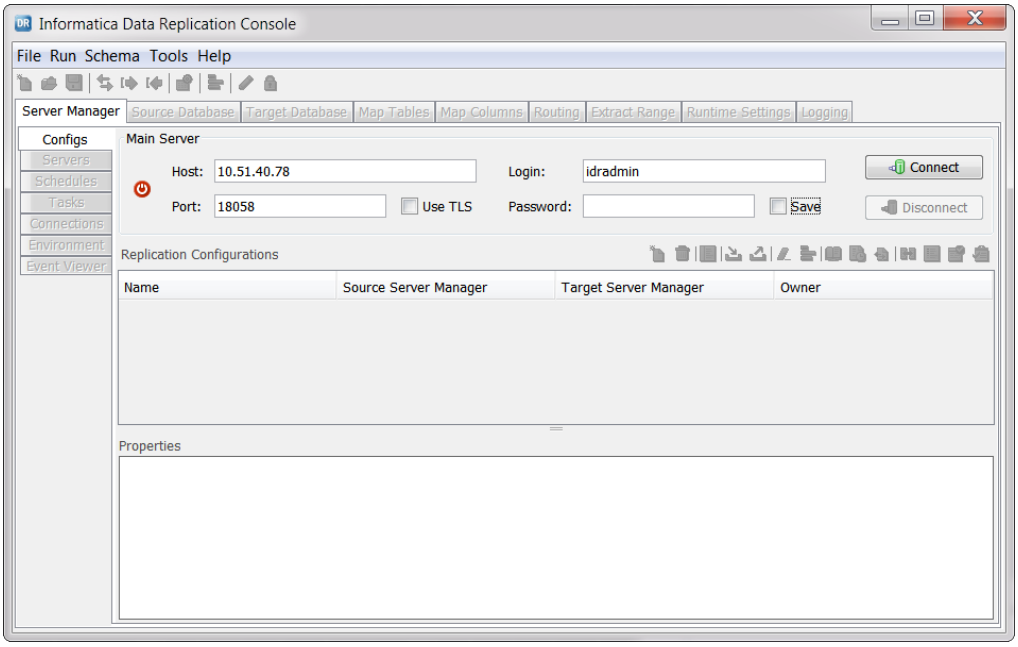
Before you begin configuring a replication job, you must define the Server Manager instances that comprise the replication server group, including the Main server and all of its subservers. You can define Server Manager instances from the Data Replication Console or from the Server Manager CLI. For more information about defining the Server Manager instances from the Server Manager CLI, see the *Informatica Data Replication Command Line Interface for the Server Manager*.

Connecting to the Server Manager Main Server

You must connect to the Server Manager Main server before running Server Manager tasks.

1. Click the **Server Manager** tab > **Configs** view.

   The following image shows this view:



2. Enter connection information for the Server Manager Main server in the following fields:

   Host

   Host name or IP address of the system with the Server Manager Main server.
   Informatica recommends that you use the actual host name or IP address to connect to the Server Manager Main server. For a replication topology with multiple Server Manager instances, if you use "localhost" as the host name or use 127.0.0.1 as the IP address for the Server Manager Main server, subservers cannot access the Main server.

   Port

   Port number. Valid values are integers from 1 through 65535. The default port number is 8088.

   Use TLS

   Select this check box if the Server Manager Main server uses HTTPS connections.

   Login

   User name.
   The first time you connect to the Server Manager Main server, you must log in as the idradmin user. By default, the idradmin user has no password.

   Password

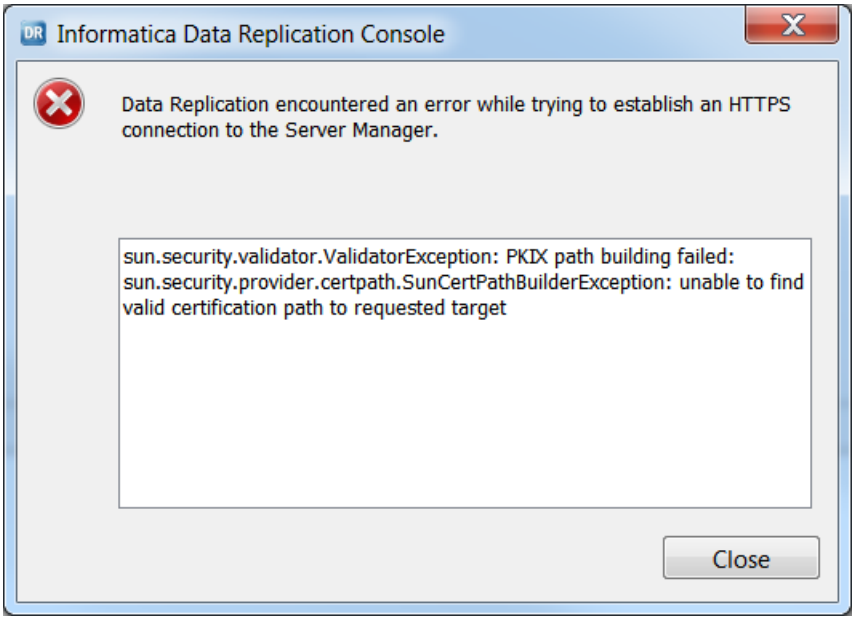
   User password.

3. Optionally, select the **Save** check box to save the password.
4. Click **Connect** to connect to the Main server.

Under **Replication Configurations**, a list of configurations appears. Under **Properties**, the list of general properties for the selected configuration appears.

Alternatively, if you connect to a Server Manager that uses a self-signed certificate to accept HTTPS connections for the first time, Data Replication searches for the TrustStore file that Java uses to store trusted certificates. Data Replication searches the default location of the Java TrustStore file, which is either *JAVA_HOME*/jre/lib/security/cacerts for the JDK or *JAVA_HOME*/lib/security/cacerts for the JRE.

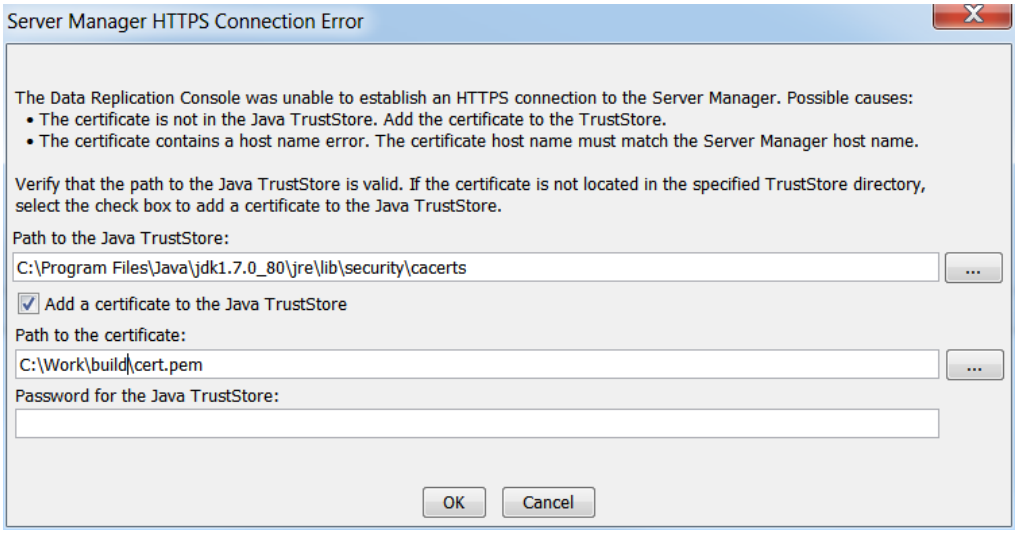
If Data Replication cannot locate the Java TrustStore file in the default directory, the following error message appears:



If your password does not meet the password-strength requirements, the Data Replication Console opens a dialog box where you can change your password to meet the requirements.

5. If the error message for a failed HTTPS connection to the Server Manager is displayed, click **Close**.

   The **Server Manager HTTPS Connection Error** dialog box appears.



6. In the **Server Manager HTTPS Connection Error** dialog box, enter or browse to a valid Java TrustStore directory in the **Path to the Java TrustStore** field.

If the certificate is not located in the specified TrustStore directory, select the **Add a certificate to the Java TrustStore** option and perform the following substeps:

a. Copy the cert.pem file from the Data Replication installation directory on the system that runs the Server Manager Main server to the computer that runs the Data Replication Console.

b. Open the copy of the cert.pem file in a text editor and delete the private key lines.

   Delete the lines from `BEGIN PRIVATE KEY` to `END PRIVATE KEY`.

   Important: The host name or IP address in the cert.pem file must match the host name or IP address that you entered on the **Server Manager** tab > **Configs** view to connect to the Server Manager Main server. To edit the host name or IP address in the cert.pem file, follow the steps for regenerating the certificate in "Configuring the Server Manager for HTTPS Communication" on page 155.

c. Enter the Java TrustStore information in the following fields:

   - **Path to the certificate**. The full path to the copy of the cert.pem file that includes only the certificate part of the original file.
   - **Password for the Java TrustStore**. The password for the Java TrustStore.

     Tip: The default password is "changeit."

d. Click **OK**.

Tip: Alternatively, you can add the certificate to the Java TrustStore manually. Run the following command:

```
keytool -import -noprompt  -trustcacerts -alias HOST_NAME_OR_IP -file
PATH_TO_CERTIFICATE_FILE -keystore PATH_TO_JAVA_TRUSTSTORE -storepass
PASSWORD_FOR_TRUSTSTORE
```

The specified *HOST_NAME_OR_IP* address in the command must match the host name or IP address that you entered on the **Server Manager** > **Configs** view to connect to the Server Manager Main server.

If the connection succeeds, the Main server appears in the **Servers** view and the other views of the **Server Manager** tab become active.

Note: If Data Replication is configured to enforce password-strength policies, the second time you connect to the Server Manager Main server as the idradmin user using the empty password, the Data Replication Console requires you to change the password.

Configuring an Environment Variables List for the Main Server

By default, Data Replication uses the system environment variables for all servers. You can optionally add a custom environment variables list to use with a particular Server Manager Main server and replication configuration.

Tip: To view the system environment variables from the **Server Manager** tab > **Servers** view , click the **System Environment** icon button on the Server Managers toolbar.

1. To add a custom environment variables list, click the **Server Manager** tab > **Environment** view.
2. Click **Custom Environment**.
3. In the **Custom Environment Editor** dialog box, click **New**.
4. In the **New** dialog box, enter a name and owner for the environment variables list and click **OK**.

   Note: The names of environment variables lists can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates list names that are longer than 100 characters.

   Only the owner of the custom environment variables list can edit it later.

5. Click **Close**.

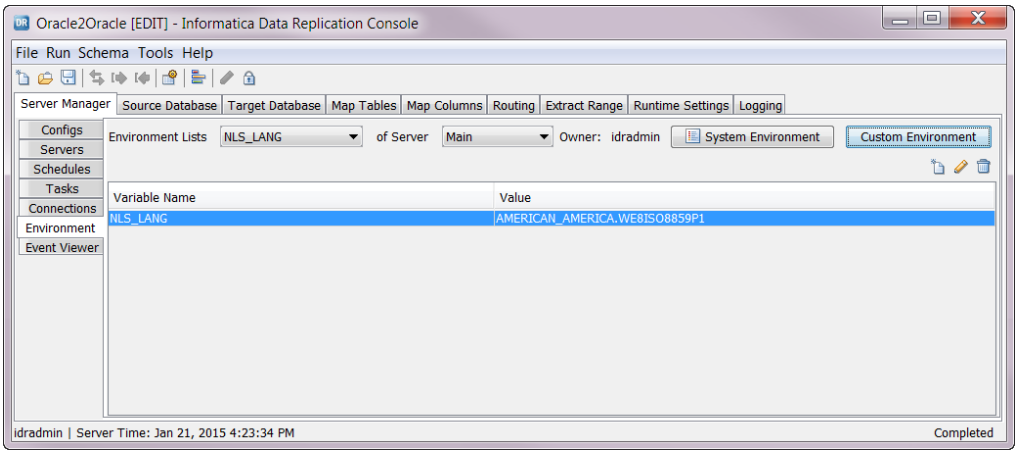
   The **Server Manager** tab > **Environment** view is displayed again.

6. In the **Environment Lists** list, select the name of the new environment variables list. Also verify that the correct Main server is selected.
7. To add an environment variable to the list, click the **New** icon button on the Server Managers toolbar.
8. In the **New** dialog box, enter the environment variable name and value. Then click **OK**.

   Note: Environment variable names can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates environment variable names that are longer than 100 characters.

   The environment variable appears in the variables list box.

   The following image shows a list with one environment variable:



   Repeat steps 7 and 8 to add more environment variables to the list.

9. To assign the custom environment variables list to a specific configuration and Server Manager Main server, perform the following substeps:

   a. Click the **Configs** view.

   b. Select a configuration and click the **Properties** icon button.

   c. In the **Runtime environment** list for the source or target Server Manager, select the environment variables list.

   At run time, Data Replication uses the environment variables that are defined in the list for replication jobs that use the configuration.

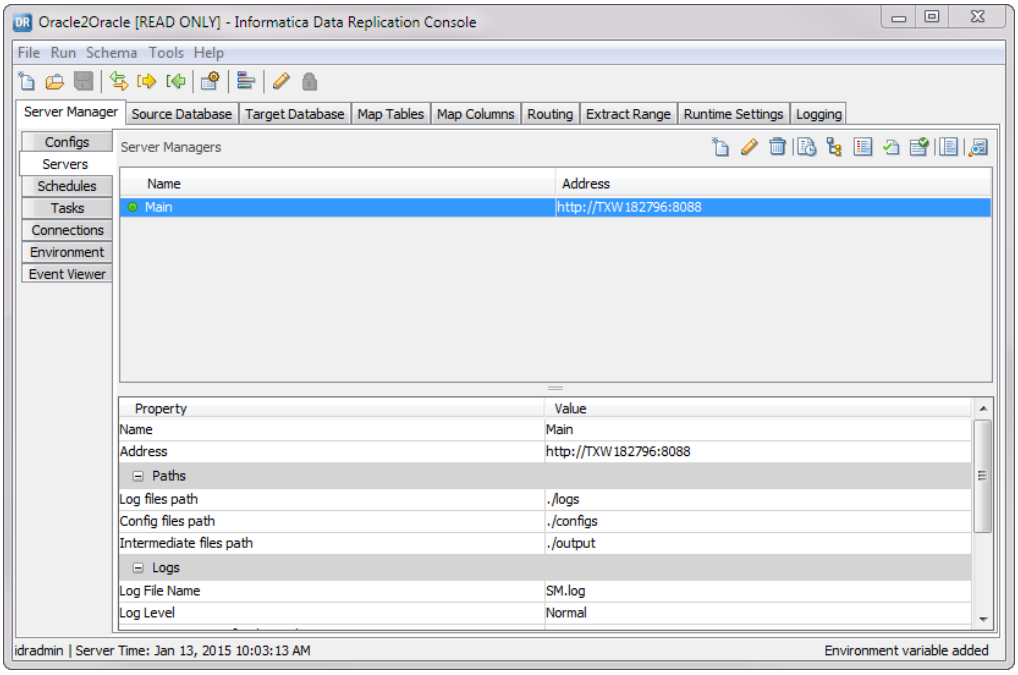
Adding Subservers

The idradmin user can add multiple subservers to the Server Manager Main server. Typically, subservers are located on remote machines where the Extractor or Applier run. The subservers run the replication tasks and transmit the intermediate files.

In the Data Replication Console, the idradmin user can either manually enter subserver connection information or discover the available servers.

1. On the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user.
2. Click the **Server Manager** tab > **Servers** view.

The following image shows this view:



A green icon to the left of the server name indicates that the server is running. A red icon indicates that the server is not available.

3. To manually add a subserver, perform the following steps:
   a. Click the **New** icon button on the Server Managers toolbar.
   b. In the **New** dialog box, enter a subserver name, host http address or URL, and port number.

      Notes:

      - Subserver names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates subserver names that are longer than 100 characters.
      - Valid port numbers are integers from 1 through 65535.

   c. If the subserver uses HTTPS connections, select **Use TLS**.
   d. To test the connection, click **Test**.
   e. If the test is successful, click **OK**.

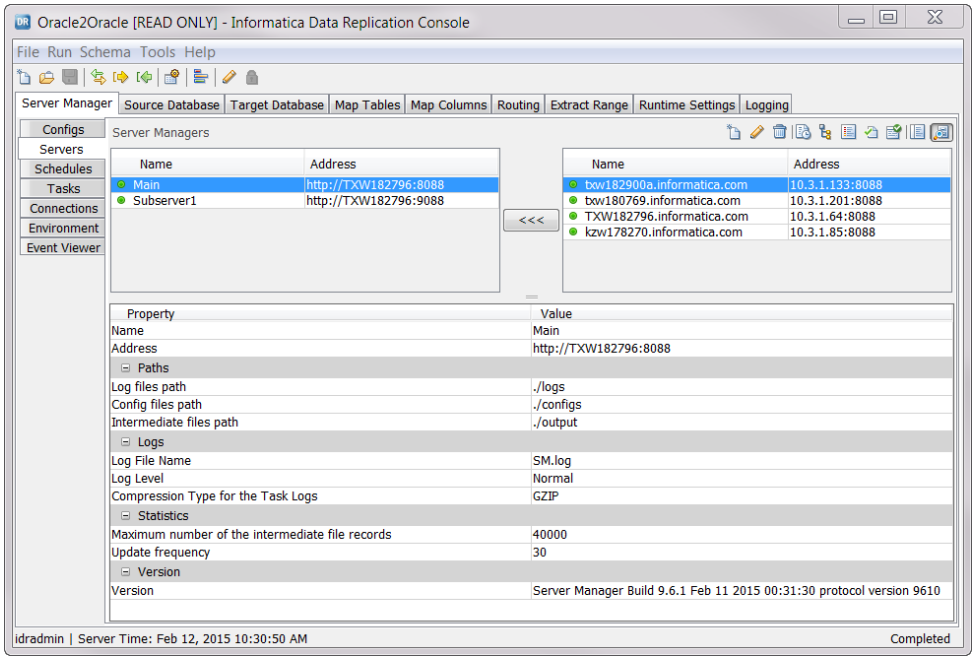
      The **Network Test** window appears. This window shows connectivity status for each combination of Server Manager instances. Statuses are: success and failed.

   f. Review the connectivity information and click **Close**.

      The subserver appears in the **Servers** view.

4. To select a subserver from a list of discovered servers, perform the following steps:
   a. Click the **Discover** icon button on the Server Managers toolbar to display a list of available servers.

      The following image shows a list of discovered servers in the right list box:



   b. Select a server in the list box on the right, and then click the left arrow button to add the server to the list box on the left.

      Note: You cannot add a subserver that is already associated with another replication server group.

      A dialog box displays subserver connection information.

   c. Verify that the connection information for the subserver is correct. If the subserver uses HTTPS connections, select **Use TLS**.

   d. Click **OK**.

      The **Network Test** window appears. This window shows the connectivity status for each combination of Server Manager instances. Statuses are: success and failed.

   e. Review the connectivity information and click **Close**.

      Tip: Click the **Hide** icon button to minimize the list of available servers.

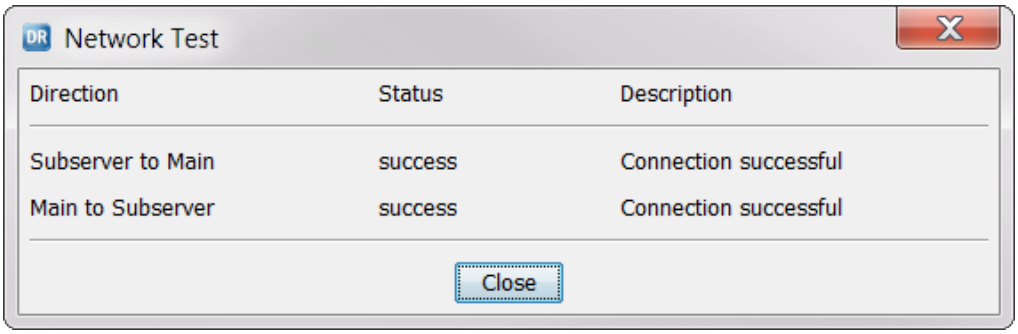
Testing Connectivity Between Server Manager Instances

If a replication configuration uses multiple Server Manager instances, you can test whether each Server Manager instance can communicate with the other Server Manager instances in the same replication server group over the network. This test can help avoid connectivity problems during data replication.

You can test if the Main server can connect to each subserver, and test if each subserver can connect to the Main server and each of the other subservers.

1. In the Data Replication Console, on the **Server Manager** tab > **Servers** view, click the **Network Test** icon above the list of Server Manager instances, or right-click a Server Manager instance and click **Network Test**.

   The **Network Test** dialog box appears.



   The **Network Test** dialog box shows the connectivity status for each combination of Server Manager instances. Statuses are: success and failed.

   Note: If the Server Manager Main server is secured with Network Address Translation (NAT), the Server Manager subservers might fail to connect to the Main server. In this case, all of the Server Manager instances must use unidirectional connections.

2. Review the connectivity information. Then click **Close**.

Related Topics:

- " Configuring the Server Manager Main Server to Run with NAT" on page 156

Editing Connection Information for a Main Server or Subserver

The idradmin user can edit the host name or address, port number, server name, and **Use TLS** setting for a Server Manager Main server or subserver.

You must edit the host information if you change the host name or IP address of the computer where the server runs or if you move the server to another computer. Ensure that the host name or IP address that you enter for the Server Manager server matches the host name or IP address of the computer where the server runs.

1. On the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user.
2. Click the **Server Manager** tab > **Servers** view.
3. Right-click the server name and click **Edit**, or click the **Edit** icon button on the Server Managers toolbar.

   The **Editing** dialog box appears.

4. Edit the information for the server in any of the following fields:
   - **Name**. The user-defined server name. Server names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates server names that are longer than 100 characters.
   - **Host**. The host name or IP address.
   - **Port**. The port number. Valid values are integers from 1 through 65535.
   - **Use TLS**. An option that controls whether to use HTTPS connections.
5. Click **Test** to test the new server connection.

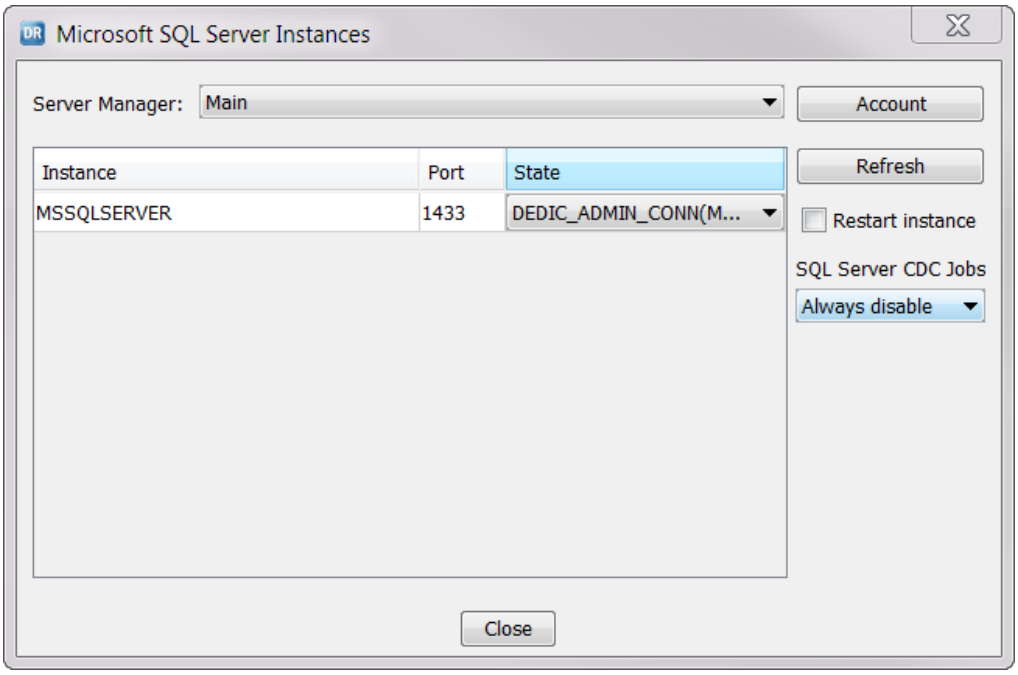
   If the test is successful, click **OK**.
6. Click **Save**.

Editing Microsoft SQL Server Instance Settings

If you run the Server Manager on a local Microsoft SQL Server source system, you can edit some connection settings for SQL Server instances. You can also indicate whether to restart SQL Server Enterprise Edition instances when you edit SQL Server Change Data Capture settings for source tables and indicate how to handle SQL Server CDC capture and cleanup jobs.

1. On the **Server Manager** tab > **Servers** view, select a Server Manager instance that runs on the SQL Server source system. Then click the **SQL Server Instances** icon on the Server Managers toolbar, or right-click the SQL Server source system and select **SQL Server Instances**.

   The **Microsoft SQL Server Instances** dialog box appears.



2. In the **Server Manager** list, verify that the Server Manager that is on the SQL Server source system is selected.
3. In the **Instance** column, select the SQL Server instance for which you want to edit settings.

Tip: To update the list of SQL Server instances, click **Refresh**.

4. In the **State** column, select the **DEDIC_ADMIN_CONN(MULTI)** or **DEDIC_ADMIN_CONN(SINGLE)** option to indicate the dedicated administrator connection state.

   A dedicated administrator connection provides administrator access to a running instance of the SQL Server Database Engine and supports encryption and other security features of Microsoft SQL Server. The Server Manager requires the dedicated administrator connection to enable or disable SQL Server Change Data Capture and to get the Change Data Capture status for each table.
   Options are:

   - **NOT_RUNNING**. The SQL Server instance is inactive.
   - **RUNNING**. The SQL Server instance runs without a dedicated administrator connection. Microsoft SQL Server Standard Edition instances use this option by default. The Data Replication Console cannot get the Change Data Capture status from instances that are running in this mode.
   - **DEDIC_ADMIN_CONN(MULTI)**. The SQL Server instance runs with a dedicated administrator connection in multiple-user mode. Microsoft SQL Server Enterprise Edition instances use this option by default. Informatica recommends that you use this option.
   - **DEDIC_ADMIN_CONN(SINGLE)**. The SQL Server instance runs with a dedicated administrator connection in single-user mode.

5. Verify that the connection information that the Server Manager uses to connect to the SQL Server instance is correct:

   a. Click the **Account** button.

      The **Microsoft SQL Server Account** dialog box appears.

   b. Verify or edit the connection information.

      The following table describes the connection fields:

Field	Description
Use Windows account	If the Server Manager runs under a Windows user account that does not have sufficient privileges to enable or disable SQL Server Change Data Capture, clear this check box and complete the **Login** and **Password** fields.
Login	If you clear the **Use Windows account** option, enter a user name that has privileges to enable or disable Change Data Capture.
Password	If you clear the **Use Windows account** option, enter a valid password for the specified user.
Use default	To use the default SQL Server Native Client driver connection string that is initially shown in the **Custom connection string** or **DAC Custom connection string** fields to connect to the SQL Server instance and get the SQL Server Change Data Capture status of each database, select this check box. To specify another connection string in either the **Custom connection string** or **DAC Custom connection string** field, clear this check box.

Field	Description
Custom connection string	Enter a connection string that the Server Manager uses to connect to the Microsoft SQL Server instance. The Server Manager also uses this connection string to enable Change Data Capture for Microsoft SQL Server Enterprise Edition sources.

DAC Custom connection string	Enter a dedicated administrator connection (DAC) string that the Server Manager uses to get the Change Data Capture status for each database. The Server Manager also uses this connection string to enable Change Data Capture for Microsoft SQL Server Standard Edition sources in single-user mode.
	For Microsoft SQL Server Cluster sources, provide the DAC string in the following format:
	`DRIVER=SQL_Server_ODBC_driver;Server=virtual_IP;`
	`Network=DBMSSOCN;Trusted_Connection=Yes;Port=port;`
	`Address=virtual_IP,port;`

c. Click **Save**.

6. For Microsoft SQL Server Enterprise Edition sources, use the **Restart instance** check box to indicate whether the Server Manager restarts the SQL Server instance in single-user mode to enable a dedicated administrator connection to the source database when you edit the Change Data Capture settings for source tables on the **Map Tables** tab.

   - Select the check box to restart the SQL Server instance in single-user mode. SQL Server does not generate the SQL Server CDC capture tables. Data Replication does not require these tables for replication processing.
   - Do not select the check box to run the SQL Server instance in multiple-user mode. If the Server Manager runs on the same system as the SQL Server instance, you do not need to restart the SQL Server instance when you edit the Change Data Capture settings for source tables on the **Map Tables** tab.

   By default, the check box is not selected.

   Notes:

   - If the list of instances in the **Microsoft SQL Server Instances** dialog box contains only SQL Server Standard Edition instances, the **Restart instance** check box is selected and unavailable for editing.
   - For all listed SQL Server Standard Edition instances, the Server Manager must restart the SQL Server instance in single-user mode to enable a dedicated administrator connection to the source database when you edit the Change Data Capture settings for source tables on the **Map Tables** tab.
   - If the **Restart instance** check box is selected, and you edit the Change Data Capture settings for source tables when multiple SQL Server sources are running on the source system, Data Replication might display an error. To avoid this error, Informatica recommends that you ensure that only the SQL Server Database Engine service is running before editing Change Data Capture settings.
   - The **Restart instance** check box is equivalent to the Server Manager RestartMSSQLInstance advanced property. When you edit either the **Restart instance** check box or the Server Manager property for Microsoft SQL Server Enterprise Edition sources, the other value is updated to the equivalent setting. For more information, see "Editing Properties for the Main Server or a Subserver" on page 138.

7. In the **SQL Server CDC Jobs** list, select one of the following options to indicate how to handle the CDC capture and cleanup jobs that SQL Server generates for databases with Change Data Capture enabled:

   - **Never disable**. Do not disable the SQL Server CDC jobs. Select this option if you do not want Data Replication to disable the SQL Server CDC jobs because the jobs are already disabled from Microsoft SQL Server Management Studio or you accept that Data Replication might not capture all of the change data. This option is the default option.

- **Disable only if CDC is not enabled**. Disable the SQL Server CDC jobs if SQL Server Change Data Capture is not enabled for any source tables when you create the configuration in Data Replication. If SQL Server Change Data Capture is enabled for any source table, the SQL Server CDC jobs are not disabled.
- **Always disable**. Disable the SQL Server CDC jobs regardless of whether SQL Server Change Data Capture is enabled or not enabled for the source tables.

Default value is **Never disable**.

Important: If you select **Never disable** or **Disable only if CDC is not enabled** and the Server Manager does not disable the SQL Server CDC jobs, disable Data Replication management of the secondary truncation checkpoint on the **Runtime Settings** tab > **General** view. Also, ensure that the Extractor reads both the transaction and backup logs. Otherwise, Data Replication might not capture all of the change data.

Note: The **SQL Server CDC Jobs** options are equivalent to the valid values for the Server Manager CdcJobsDisableMode advanced property. When you edit either the **SQL Server CDC Jobs** option or the Server Manager property, the other value is updated to the equivalent setting. For more information, see "Editing Properties for the Main Server or a Subserver" on page 138.

8. Click **Close**.

Related Topics:

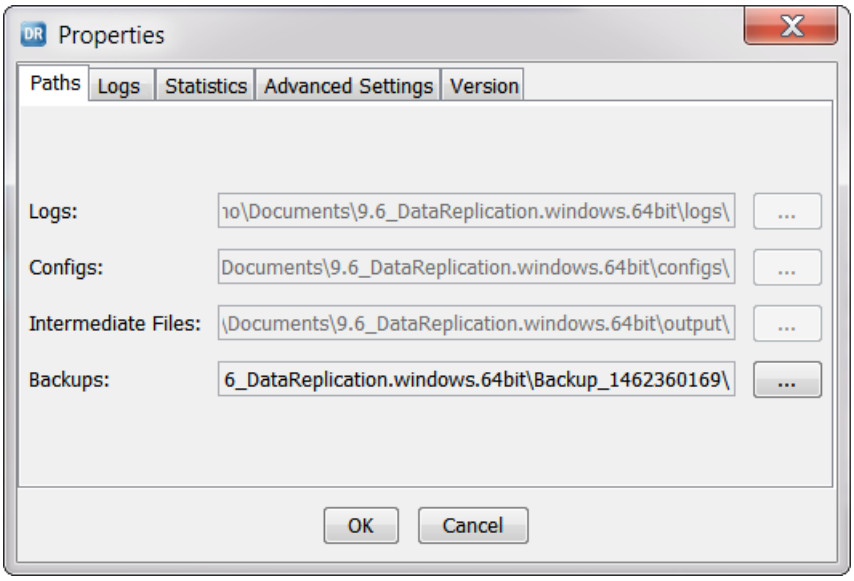
- "Replication Considerations for Microsoft SQL Server Sources" on page 68
- "Supported DDL Operations for Microsoft SQL Server Sources" on page 73
- "Managing Microsoft SQL Server Change Data Capture Settings" on page 344
- " Coordination with Microsoft SQL Server Change Data Capture" on page 70

Editing Properties for the Main Server or a Subserver

From the Data Replication Console, the idradmin user can optionally edit properties of the Server Manager Main server or a subserver, such as the locations of certain file types and the log file name and logging level.

1. On the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user.
2. On the **Server Manager** tab > **Servers** view, select a server.
3. Click the **Properties** icon button on the Server Managers toolbar, or right-click the server row and click **Properties**.

The **Properties** dialog box appears.



4. On the **Paths** tab, change the locations of log files, configuration files, intermediate files, and backups of configuration SQLite database files and intermediate files for configurations that have undergone a Clean operation. Click the **Browse** button next to a field to browse to another location.

   Ensure that the path to a new location does not include space characters.

   Note: Do not specify a Network File System (NFS) location for the configuration files. SQLite databases on NFS do not support access by multiple Data Replication processes at the same time. Access by multiple processes might cause incorrect behavior of the replication components.

   After you change a location for a file type, stop the Server Manager and copy any existing files to the new location manually. The Server Manager does not move these files to the new location. Even if you do not have existing files to copy to the new location, you must restart the Server Manager to apply the changes to file locations.

5. On the **Logs** tab, change the log file base name, logging level, and log compression setting for the selected server.

   You can use GZIP or ZIP as the compression method, or turn off compression of the log file.

6. On the **Statistics** tab, specify settings for storing and collecting replication statistics.

   You can edit the following properties that appear by default:

   Maximum number of the intermediate file records

   The maximum number of records with metadata for the processed intermediate files that Data Replication can store in the Extractor SQLite database and the buffer SQLite database. The Server Manager uses the buffer SQLite database to pass data to the Applier. When the number of intermediate-file records exceeds this maximum, the Server Manager deletes records down to the maximum limit, starting with the records for the earliest intermediate files. Default is 40000 records.

   Update frequency

   The number of milliseconds that must elapse before Data Replication updates replication statistics. This value determines the frequency at which Data Replication transmits statistical data from Server Manager subservers to the Server Manager Main server. Default is 30 milliseconds, which is equivalent to 33 updates per second.

7. On the **Advanced Settings** tab, edit advanced properties for the selected Server Manager server.

Edit any of the following properties:

ApiThreadsCount

If the DynamicThreadAllocation property of the Server Manager is set to 0, specifies the number of threads that the Server Manager uses for processing incoming requests from other Server Manager instances in unidirectional mode.

If the DynamicThreadAllocation property of the Server Manager is set to 1, the Server Manager uses this property for the initial allocation of threads and then dynamically adjusts the thread count.

Valid values: 1 through 1024.

Default: 10.

ApproximateCalculationStat

Determines the accuracy level for replication statistics. This setting affects the end-to-end replication latency that is displayed in the Statistics windows. Options are:

- **0**. Provide replication statistics that have a higher level of accuracy but that are updated less frequently. When calculating replication statistics, the Server Manager analyzes only the intermediate files that contain committed transactions. If an intermediate file contains at least one open transaction, the Server Manager waits to get the intermediate file until the transaction commits and then calculates the statistical data for all transactions in the intermediate file. This processing can cause delays in getting the statistical data.
- **1**. Provide approximated replication statistics. When calculating the replication statistics, the Server Manager analyzes all of the intermediate files that the Applier processed. The intermediate files might contain open transactions. Use this option to avoid delays in getting the statistical data related to large transactions.

Default: 1.

After you edit this property value, you must restart the Server Manager.

AsyncHttpEnabled

Determines whether the Server Manager uses bidirectional or unidirectional connections. Options are:

- **0**. Use bidirectional connections between the Server Manager Main server and its subservers.
- **1**. Use unidirectional connections between the Server Manager Main server and its subservers. Unidirectional connections are slower than bidirectional connections. Use this option only if you deploy the Server Manager Main server in a local area network with NAT.

Default: 0.

BodyResponseTimeout

Specifies the number of seconds that a Server Manager subserver must wait for a response from the Server Manager Main server in the unidirectional mode.
Valid values: 0 through 2147483647 seconds.

Default: 60 seconds.

BuiltinTaskManagersCountThreads

If the DynamicThreadAllocation property of the Server Manager is set to 0, specifies the number of threads for running replication tasks on one Server Manager. Each thread can run a single replication task.

If the DynamicThreadAllocation property of the Server Manager is set to 1, this property is used for initial allocation of threads. If necessary, the Server Manager can create more threads and then close threads after they complete their tasks.

Valid values: 1 through 1024.

Default: 20.

CdcJobsDisableMode

For configurations with Microsoft SQL Server sources, determines whether the Server Manager disables the SQL Server [cdc].*database_name*_capture job and [cdc].*database_name*_cleanup job after enabling SQL Server Change Data Capture for the source database and tables. Options are:

- **0**. Do not disable the SQL Server CDC jobs. Select this option if you do not want Data Replication to disable the CDC jobs because the jobs have already been disabled from Microsoft SQL Server Management Studio or you accept that Data Replication might not capture all of the change data.
- **1**. Disable the SQL Server CDC jobs if SQL Server Change Data Capture is not enabled for any source tables when you create the configuration in Data Replication. If SQL Server Change Data Capture is enabled for any source table, the SQL Server CDC jobs are not disabled.
- **2**. Disable the SQL Server CDC jobs regardless of whether SQL Server Change Data Capture is enabled or not enabled for the source tables.

Default: 0.

Important: If you select **0** or **1** and the Server Manager does not disable the SQL Server CDC jobs, disable Data Replication management of the secondary truncation checkpoint on the **Runtime Settings** tab > **General** view. Also, ensure that the Extractor reads both the transaction and backup logs. Otherwise, Data Replication might not capture all of the change data.

Note: The options for this property are equivalent to the options in the **SQL Server CDC Jobs** list in the **Microsoft SQL Server Instances** dialog box. When you edit either this property or the **SQL Server CDC Jobs** setting, the other value is updated to the equivalent setting. For more information, see "Editing Microsoft SQL Server Instance Settings" on page 135.

CheckPasswordLength

Main server only. Specifies the minimum number of characters that passwords for user accounts must contain. If this property is set to 0, the Data Replication Console and the Server Manager Command-Line Interface (CLI) do not enforce a minimum password length.

Valid values: 0 through 2147483647.

Default: 0.

CountDaysOfStorageHistory

Determines the number of days to store replication log data. This data includes the following types of information:

- Statistical data on Extractor performance, which is displayed on the **Extractor Performance** tab of the Statistics window.
- Data for Server Manager events related to schedules and tasks, which is displayed on the **Server Manager** tab > **Logs** view > **Journal** tab.

Valid values: 0 through 2147483647 days.

Default: 30 days.

CountDaysofStorageSMLogs

Determines the number of days to store Server Manager log data.

Valid values: 1 through 60 days.

Default: 30 days.

CountThreads

Specifies the number of active threads for receiving messages.
After you edit this property value, you must restart the Server Manager.

Valid values: 1 through 1024.

Default: 50.

After you edit this property value, you must restart the Server Manager.

DBPasswordEncryptionEnabled

Main server only. Indicates whether Data Replication encrypts the database passwords that you specify for the source and target database connections. Options are:

- **0**. Do not encrypt database passwords.
- **1**. Encrypt database passwords by using the AES256 encryption algorithm.

  Important: If you did not add the Java Cryptography Extension (JCE) files to the *JRE_HOME*/bin/security folder, the Data Replication Console does not allow you to set this property to 1.

Default: 0.

DisableStatistics

Determines whether the Server Manager collects statistics on the latency of the Extractor, Applier, and Send File tasks by heartbeat for continuous replication. Options are:

- **0**. Collect latency statistics.
- **1**. Do not collect latency statistics.

Default: 0.

DynamicThreadAllocation

Determines whether the Server Manager dynamically allocates threads for running replication tasks, processing incoming requests, and sending messages to other Server Manager instances. Options are:

- **0**. Do not allocate Server Manager threads dynamically. Use the ApiThreadsCount, BuiltinTaskManagersCountThreads, IntermediateFileTransfersCountThreads, and SendCountThreads properties to specify the number of active threads that the Server Manager allocates to these tasks.
- **1**. Allocate Server Manager threads dynamically. The Server Manager uses the values of the ApiThreadsCount, BuiltinTaskManagersCountThreads, IntermediateFileTransfersCountThreads, and SendCountThreads properties for initial allocation of threads. If necessary, the Server Manager can create more threads and then close threads after they complete their tasks.

Default: 1.

ExpirationPeriodForPasswords

Main server only. Specifies the number of days after which passwords for user accounts expire. When a password for a user account expires, the Server Manager Main server stops responding to user requests from the Data Replication Console. After the Server Manager Main server refuses a user request, the Data Replication Console or the Server Manager Command-Line Interface (CLI) in interactive mode prompts the user to change the expired password.

Valid values: 0 through 24000 days. To disable password expiration, specify 0.

Default: 0.

FileAndLineLogInfoEnable

Determines whether the Server Manager includes the source code file name and line number in Server Manager logs. The file name and line information indicate the location in the source code from which the Server Manager generated the logs. Options are:

- **0**. Do not include the file name and line information in Server Manager logs.
- **1**. Include the file name and line information in Server Manager logs.

Default: 0.

Generate Internal Master Key

Main server only. Indicates whether the Server Manager Main server can generate an internal master key to encode master keys from an Oracle TDE wallet. Options are:

- **0**. Do not generate an internal master key.
- **1**. Generate an internal master key if one is not available in the directory that is specified in the **Internal Master Key Path** property.

Default: 1.

HeaderResponseTimeout

Specifies the number of seconds that a Server Manager subserver must wait for the Server Manager Main server to retrieve a request in unidirectional mode.
Valid values: 0 through 2147483647 seconds.

Default: 5 seconds.

HTTP

Indicates whether to enable HTTP communication for the Server Manager. Options are:

- **0**. Do not use HTTP connections for the Server Manager.
- **1**. Use HTTP connections for the Server Manager.

Default: 1.

After you edit this property value, restart the Server Manager. This property is a synonym for the http command line parameter for the Server Manager.

HttpCharsetEncoding

Specifies the character encoding of the system console that runs the Server Manager.
Default: UTF-8.

HttpCurlDebug

Indicates whether the libcurl library logs debug information about HTTP requests and responses between the Data Replication Console and Server Manager instances. This information is added to the Server Manager logs. Options are:

- **0**. Do not log libcurl debug messages about HTTP requests and responses.
- **1**. Log libcurl debug messages about HTTP requests and responses. Set this property to 1 only at the request of Informatica Global Customer Support.

Default: 0.

HttpCurlVerbose

Indicates whether the libcurl library logs verbose information about HTTP requests and responses between the Data Replication Console and Server Manager instances. This information is added to the Server Manager logs. Options are:

- **0**. Do not log libcurl verbose messages about HTTP requests and responses.
- **1**. Log libcurl verbose messages about HTTP requests and responses. Set this property to 1 only at the request of Informatica Global Customer Support.

Default: 0.

HttpKeepAlive

Indicates whether the Server Manager uses keep-alive sessions. Options are:

- **1**. Enable keep-alive sessions.
- **0**. Disable keep-alive sessions.

Default: 1.

After you edit this property value, you must restart the Server Manager.

HttpKeepAliveTimeout

If the Server Manager HttpKeepAlive advanced property is set to 1, specifies the amount of time, in milliseconds, that a TCP/IP keep-alive session that the Server Manager uses can have no network traffic before the session times out.
Valid values: 0 through 2147483647 milliseconds.

Default: 10000 milliseconds.

HTTPS

Indicates whether to enable HTTPS communications for the Server Manager. Options are:

- **0**. Do not use HTTPS connections for the Server Manager.
- **1**. Use HTTPS connections for the Server Manager.

Default: 0.

After you edit this property value, restart the Server Manager. This property is a synonym for the https command line parameter for the Server Manager.

HttpsVerifyHost

For HTTP and HTTPS protocols, indicates whether the Server Manager uses the libcurl library to verify that the machine that hosts the database server has the same name in the certificate as in the URL. Options are:

- **0**. Disable host verification.
- **1**. Enable host verification.

Default: 0.

HttpsVerifyPeer

For HTTP and HTTPS protocols, indicates whether the Server Manager uses the curl open source tool and library to verify the authenticity of a peer certificate. When negotiating a TLS or SSL connection to the Server Manager, the server sends a certificate that identifies itself for verification. Options are:

- **0**. Disable peer verification.
- **1**. Enable peer verification.

Default: 0.

IFViewersMaxBlobSize

Specifies the maximum size, kilobytes, of the LOB data that the Data Replication Console displays in the **Intermediate File Information** dialog box for the before- and after-image values in an intermediate file. If the size of the LOB data in the intermediate file exceeds the specified value, the Data Replication Console displays a truncated LOB value.

When determining this value, consider the amount of memory that you allocated for the Java Virtual Machine (JVM.) The Data Replication Console might hang when allocating space for multiple large LOB values in the intermediate file.

Valid values: 1 through 2147483647 KB.

Default: 1 KB.

IntermediateFilesDeleteLatency

Determines the period of time, in seconds, that transaction-related statistics for intermediate files are available for viewing in the Data Replication Console after the Applier processes the files.
Valid values: 0 through 2147483647 seconds.

Default: 60 seconds.

IntermediateFileTransfersCountThreads

If the DynamicThreadAllocation property of the Server Manager is set to 0, specifies the number of threads that the Server Manager uses for running Send File tasks on one Server Manager.

Valid values: 1 through 1024.

Default: 10.

If the DynamicThreadAllocation property of the Server Manager is set to the default value of 1, the Server Manager uses this property for initial allocation of threads and then dynamically adjusts the thread count.

Internal Master Key Path

Specifies the path to the internal master key that Server Manager instances use to encode master keys from an Oracle TDE wallet. The Server Manager Main server can generate an internal master key if the GenerateInternalMasterKey property is set to 1.

Default: ./idrkey.

After you edit this property value, you must restart the Server Manager.

LimitValueFreeSpace

Main server only. Specifies the minimum amount of available disk space, in gigabytes, required for the directories that the Server Manager Main server uses. These directories are logs, configs, output, and DBSYNC_HOME. When the amount of available disk space for these directories is less than this property value, the Server Manager is disabled. To enable the Server Manager again, create at least the amount of available disk space that is specified in the ThresholdValueFreeSpace property for each directory that the Server Manager Main server uses.
Valid values: 0 through 2147483647 GB.

Default: 1 GB.

LogBytesToRead

If you select an event message in the **Event Viewer** view and click **Show Log**, controls how many bytes of information that the Event Viewer reads from the Server Manager log and displays in the

Log dialog box, after the selected message. You can use this property to limit the amount of log information that is displayed when you are trying to diagnose an error.
Valid values: 200 through 1000000 bytes.

Default: 2000 bytes.

LoginBlockTimeout

Main server only. Specifies the duration, in seconds, that a user account is locked out from logging in to the Server Manager Main server after the maximum number of failed login attempts that is specified in the MaxLoginAttempts property is reached.

Note: The idradmin user can unlock an account. To unlock an account, click **File** > **Users** and then click **Unlock Account** in the **Users** dialog box.

Valid values: 1 through 2147483647 seconds.

Default: 1800 seconds (30 minutes).

LogReadIndent

If you select an event message in the **Event Viewer** view and click **Show Log**, controls how many bytes of information that the Event Viewer reads from the Server Manager log and displays in the **Log** dialog box, before the selected message. You can use this property to limit the amount of log information that is displayed when you are trying to diagnose an error.
Valid values: 50 through 1000000 bytes.

Default: 2000 bytes.

MaxChangesNumberInConfigForPatching

Controls the method of processing the configuration file. When Data Replication synchronizes the configuration file with the configuration file that is stored in the Server Manager SQLite subsystem, it calculates the difference between these files. If the change record volume is less than the maximum value that this property specifies, Data Replication processes only the change records. Otherwise, Data Replication processes the entire configuration file.
Valid values: 0 through 2147483647 rows or lines.

Default: 1000 rows or lines.

MaxLoginAttempts

Main server only. Specifies the maximum number of times that users can try to log in to the Server Manager Main server from the Data Replication Console or the Server Manager CLI. If a user fails to enter a correct password after the specified number of retries, the Server Manager locks out the user account for the duration that is specified in the LoginBlockTimeout property.

To allow users an unlimited number of login attempts, set this property to 0.

Valid values: 1 through 2147483647.

Default: 5.

MaxRequestSize

Specifies the size of the buffer, in bytes, that is allocated to each Server Manager thread for receiving messages. Increase this value if the Data Replication Console fails to save your configuration.
Valid values: 1 through 2147483647 bytes.

Default: 1048576 bytes.

After you edit this property value, you must restart the Server Manager.

MonitoringEnable

Main server only. Determines whether Data Replication monitors the amount of available disk space for the Server Manager Main server directories and the creation of a new Server Manager log file each time the number of hours specified in the PeriodForSwitchSMLog property elapses. Options are:

- **0**. Do not enable the monitoring of available disk space and the creation of new log files at the PeriodForSwitchSMLog interval.
- **1**. Enable the monitoring of available disk space and the creation of new log files at the PeriodForSwitchSMLog interval.

Default: 1.

NotDeleteIntermediateFiles

Controls whether the Server Manager automatically deletes the intermediate files after the Applier processes them. Options are:

- **0**. Delete the intermediate files.
- **1**. Do not delete the intermediate files.

Default: 0.

Number of restart attempts

Specifies the maximum number of restart attempts for replication tasks that run in continuous mode. If a replication task that runs in continuous mode fails, the Server Manager tries to restart the task at adaptive intervals that are based on the TaskMaxTimeOut and TaskResetTimeoutTime properties. After the Server Manager reaches the specified maximum number of restart attempts without restarting the task, the Server Manager reports a task failure.

Valid values: 1 through 2147483647.

Default: 3.

Number of tables per query for Microsoft SQL Server Change Data Capture

For Microsoft SQL Server Standard Edition sources, determines the maximum number of source tables for which SQL Server enables Change Data Capture in response to each Server Manager request.

Valid values: 1 through 2147483647.

Default: 10000.

After you edit this property value, you must restart the Server Manager.

PassThroughFileQueueSize

Specifies the maximum internal queue size, in kilobytes, for chunks of subserver task log files that pass through the Main server. After the log file chunks pass through the Main server, you can download the log files from the Data Replication Console. If the log file queue size is greater than this property value, the Server Manager does not place additional chunks of subserver task log files in the queue until the Server Manager has processed the accumulated chunks.
Valid values: 1 through 2097152 KB.

Default: 1000 KB.

PasswordCaseSensitive

Main server only. Indicates whether passwords for user accounts must include both lowercase and uppercase alphabetic characters. Options are:

- **0**. Passwords are not required to include both lowercase and uppercase alphabetic characters.
- **1**. Passwords must include both lowercase and uppercase alphabetic characters.

Default: 0.

PasswordContainNumbers

Main server only. Indicates whether passwords for user accounts must include one or more numbers. Options are:

- **0**. Passwords are not required to include numbers.
- **1**. Passwords must include numbers.

Default: 0.

PasswordContainSpecialCharacters

Main server only. Indicates whether passwords for user accounts must include one or more special characters. Options are:

- **0**. Passwords are not required to include special characters.
- **1**. Passwords must include at least one of the following special characters:
  - Space ( )
  - Exclamation mark (!)
  - Double quotation mark (")
  - Hash sign (#)
  - Dollar sign ($)
  - Percent sign (%)
  - Ampersand (&)
  - Single quotation mark (')
  - Left parenthesis (()
  - Right parenthesis ())
  - Asterisk (*)
  - Plus sign (+)
  - Comma (,)
  - Minus sign (-)
  - Full stop (.)
  - Slash (/)
  - Colon (:)
  - Semicolon (;)
  - Less than (<)
  - Equal sign (=)
  - Greater than (>)

- Question mark (?)
- At sign (@)
- Left bracket ([)
- Backslash (\)
- Right bracket (])
- Caret (^)
- Underscore (_)
- Grave accent (`)
- Left brace ({)
- Vertical bar (|)
- Right brace (})
- Tilde (~)

Default: 0.

PeriodForAskServersForFreeSpace

Main server only. Specifies the interval, in seconds, between each Server Manager Main server check of available disk space for the Server Manager Main server directories.
Valid values: 0 through 2147483647 seconds.

Default: 60 seconds.

PeriodForSwitchSMLog

Specifies the maximum number of hours that a Server Manager instance can write messages to its log file. When this time period elapses, the Server Manager creates a new log file.

Note: If the size of a log file exceeds the Size For Switching SM Log property value before this time period elapses, the Server Manager creates a new log file.

Valid values: 1 through 168 hours.

Default: 24 hours.

PeriodOfDumpStatistic

Main server only. Specifies the number of hours that the Server Manager Main server keeps detailed statistics. After this period elapses, the Server Manager Main server archives the data. Archived statistics are accurate to 5 seconds.
Valid values: 0 through 2147483647 hours.

Default: 24 hours.

RestartMSSQLInstance

For configurations with Microsoft SQL Server Enterprise Edition sources, determines whether the Server Manager restarts the SQL Server instance in single-user mode to enable a dedicated administrator connection to the source database when you edit the Change Data Capture settings for source tables on the **Map Tables** tab. Options are:

- **0**. The Server Manager does not restart the SQL Server instance when you change the Change Data Capture setting for a source table.
- **1**. The Server Manager restarts the SQL Server instance every time you change the Change Data Capture setting for a source table.

Default: 0.

Notes:

- For Microsoft SQL Server Standard Edition sources, the Server Manager always restarts the SQL Server instance when you change the Change Data Capture setting for a source table, regardless of the value for the RestartMSQLInstance property.
- If the RestartMSSQLInstance property is set to 1, and you edit the Change Data Capture settings for source tables when multiple SQL Server sources are running on the source system, Data Replication might display an error. To avoid this error, Informatica recommends that you ensure that only the SQL Server Database Engine service is running before editing Change Data Capture settings.
- The values for the RestartMSSQLInstance property are equivalent to the **Restart instance** check box on the **Microsoft SQL Server Instances** dialog box. When you edit either the check box selection or the Server Manager property value, the other value updates to match the new setting. For more information, see "Editing Microsoft SQL Server Instance Settings" on page 135.

Seconds of sleep between operations for Microsoft SQL Server Change Data Capture

For Microsoft SQL Server Standard sources, specifies the interval, in seconds, between Server Manager requests to SQL Server to enable Change Data Capture at the table level and at the column level. After the Server Manager sends a request to SQL Server to enable Change Data Capture at the table level, the Server Manager waits the specified number of seconds and then sends a request to SQL Server to enable Change Data Capture at the column level.

Increase this property value if you configured the Server Manager to enable Change Data Capture for a large number of source tables in a single request. If the Server Manager fails to enable Change Data Capture for a subset of tables, increase this value and try to enable Change Data Capture from the Data Replication Console again.

Valid values: 0 through 2147483647 seconds.

Default: 5 seconds.

SendCountThreads

If the DynamicThreadAllocation property of the Server Manager is set to 0, specifies the number of threads that the Server Manager uses for sending messages.

If the DynamicThreadAllocation property of the Server Manager is set to 1, the Server Manager uses this property for initial allocation of threads and then dynamically adjusts the thread count.

Valid values: 1 through 1024.

Default: 20.

SendFileWaitTimeout

Specifies the interval, in seconds, that the Send File task waits for the Extractor to notify the Server Manager that intermediate files have been transferred to the Applier. If this interval elapses without the notification being sent, the Server Manager retrieves the intermediate file information from the configuration SQLite database manually.
Valid values: 10 through 3600 seconds.

Default: 60 seconds.

SendMailTimeOut

Main server only. Specifies the interval, in minutes, that the Server Manager Main server waits after sending an email notification to users about a specific event type, such as an error, before sending another email notification for the same event type.
Valid values: 0 through 2147483647 minutes.

Default: 10 minutes.

Server ID

Specifies the unique key that identifies the Server Manager servers in a replication group of servers. The Server Manager Main Server generates keys for all of the subservers for security purposes. Do not modify this value.

ServerActivityEnable

Determines whether to enable server activity. This property is read-only. You cannot edit it in the Data Replication Console. The Server Manager determines the value of this property during the check for available disk space. If the available disk space is sufficient for the Server Manager, Data Replication selects the default value. Options are:

- **0**. Do not enable server activity.
- **1**. Enable server activity.

Default: 1.

Set NLS_LANG

Indicates whether the Server Manager sets the NLS_LANG environment variable for the Oracle Extractor and Oracle Applier tasks. The Server Manager gets the NLS_LANG value from the replication configuration that includes the character sets of the source and target databases. Options are:

- **0**. Do not set the NLS_LANG environment variable. The Extractor and Applier tasks use the local NLS_LANG environment variable that is set for the database server on the operating system.
- **1**. Set the NLS_LANG environment variable for the Extractor and Applier tasks to the values from the replication configuration.

Default: 1.

Size For Switching SM Log

Specifies the maximum size of a log file, in megabytes, to which a Server Manager instance logs messages. When this log size limit is exceeded, the Server Manager creates a new log file.

Note: If the time period that is specified in the PeriodForSwitchSMLog property elapses before this log size limit is exceeded, the Server Manager creates a new log file.

Valid values: 1 through 100 MB.

Default: 10 MB.

SmtpCurlDebug

Main server only. Indicates whether the libcurl library adds debug information about email notifications to the Server Manager Main server logs. Options are:

- **0**. Do not log libcurl debug messages about email notifications.
- **1**. Log libcurl debug messages about email notifications. Set this property to 1 only at the request of Informatica Global Customer Support.

Default: 0.

SmtpCurlVerbose

Main server only. Indicates whether the libcurl library adds verbose information about email notifications to the Server Manager Main server logs. Options are:

- **0**. Do not log libcurl verbose messages about email notifications.
- **1**. Log libcurl verbose messages about email notifications. Set this property to 1 only at the request of Informatica Global Customer Support.

Default: 0.

SQLiteBusyTimeout

Specifies the number of milliseconds that the Server Manager or a replication task can wait to open the configuration SQLite database to run a SQL statement when the database is locked by another task. When the specified time elapses and the database is still locked, the request to the SQLite database fails.

Valid values: 1 through 2147483647 milliseconds.

Default: 30000 milliseconds.

SQLite_cache_size

Controls the cache size that the SQLite subsystem uses.
Valid values: 1 through 2147483647 bytes.

Default: 16384 bytes.

SQLite_page_size

Controls the page size that the SQLite subsystem uses.
Valid values: 1 through 2147483647 bytes.

Default: 16384 bytes.

StatisticCacheSendCount

Specifies the number of statistic objects that the Server Manager sends at one time from an internal cache to the Main server. The Server Manager retrieves statistical data from intermediate files for the Extractor and Applier, stores the data in an internal cache, and then sends the data in batches to the Main server. This property regulates the amount of statistical data that Data Replication transmits between servers at one time. Use this property to limit the network load for large requests of statistical data.
Valid values: 1 through 2147483647.

Default: 100.

StopChainWhenTaskFails

Controls whether to stop the continuous schedule and all of the schedule tasks when one or more tasks in the schedule end abnormally. Options are:

- **0**. Do not stop the schedule and its running tasks. If one or more tasks in the schedule end abnormally, the remaining tasks continue to run.
- **1**. Stop the schedule and all of its running tasks.

Default: 0.

SynchronizePeriod

Main server only. If you set the UseSynchronizationTimeWithMain to synchronize clock times of the Server Manager subservers with the clock time of the Main server, specifies the number of hours that must elapse before clock synchronization occurs.
Valid values: 0 through 2147483647 hours.

Default: 1 hour.

TaskMaxTimeout

Specifies the maximum duration, in minutes, of the adaptive sleep interval for replication tasks that run in continuous mode. If the Server Manager fails to start a replication task, it attempts to restart the task after 1 second. If the restart fails, the Server Manager doubles the interval. For subsequent restart attempts, the Server Manager continues to double the interval until this specified maximum

timeout value is reached. Thereafter, the Server Manager uses this specified timeout value until the TaskResetTimeoutTime interval elapses.
Valid values: 0 through 2147483647 minutes.

Default: 5 minutes.

TaskResetTimeoutTime

Specifies the number of hours that a replication task must run in continuous mode before the Server Manager resets the adaptive sleep interval that is specified by the TaskMaxTimeOut property. If the task fails after the specified number of hours elapses, the Server Manager tries to restart the task after 1 second and thereafter at increasing intervals based on the TaskMaxTimeOut property.
Valid values: 0 through 2147483647 hours.

Default: 2 hours.

ThresholdValueFreeSpace

Main server only. Specifies the minimum amount of available disk space, in gigabytes, required for the Server Manager Main server to resume processing after it is disabled based on the LimitValueFreeSpace property value.
Valid values: 0 through 2147483647 GB.

Default: 5 GB.

TimeToStoreBackups

Specifies the number of days that the Server Manager Main server stores backups of configuration SQLite database files and intermediate files for configurations that were cleaned.

Valid values: 0 through 24000 days.

Default: 30 days.

TimeoutOfEditConfig

Specifies the maximum number of minutes that a configuration can remain locked to the Applier while the Server Manager is saving the configuration. The Server Manager locks a configuration if you allowed the tasks to keep running while the configuration is open for editing. When the specified time elapses, the Server Manager unlocks the configuration.
Valid values: 0 through 2147483647 minutes.

Default: 2 minutes.

TmpDirectory

Specifies the directory in which a Server Manager instance stores temporary service files for viewing intermediate files.

Default: *DataReplication_installation*/tmp.

TransferThreadPoolsCountThreads

Specifies the maximum number of intermediate files that are transferred simultaneously.
Valid values: 1 through 1024.

Default: 10.

After you edit this property value, you must restart the Server Manager.

UseSynchronizationTimeWithMain

Main server only. Indicates whether to synchronize the clock times of the Server Manager subservers with the clock time of the Main server. Options are:

- **1**. Disables clock synchronization. Select this option if you use the Network Time Protocol (NTP) to connect to Server Manager servers.
- **0**. Enables clock synchronization to get realistic replication statistics for the Send File task. The related SynchronizePeriod property determines the frequency of clock synchronization.

Default: 1.

WarningValueFreeSpace

Main server only. Specifies the amount of available disk space, in gigabytes, below which the Server Manager Main server sends a notification that disk space is low. The Server Manager sends this notification only if the amount of available disk space is still greater than the minimum available disk space that is specified in the LimitValueFreeSpace property.
Valid values: 0 through 2147483647 GB.

Default: 10 GB.

8. Click **Save**.

Viewing Information About the Server Manager System

From the Data Replication Console, you can view information about the system on which the Server Manager is installed. This information includes the Server Manager version and license expiration date, operating system details, and available disk space for the Data Replication directories that the Server Manager uses.

You might need to provide this information to Informatica Global Customer Support. You can also use this information to troubleshoot Server Manager errors related to insufficient disk space.

1. On the **Server Manager** tab > **Configs** view, connect to the Main server.
2. On the **Server Manager** tab > **Servers** view, select a server.
3. Click the **Properties** icon button on the Server Managers toolbar, or right-click the Server Manager row and click **Properties**.

   The **Properties** dialog box appears.
4. Click the **Version** tab.

   View the following information about the Server Manager system:

Server Manager Version

The Server Manager version installed on the system, and the Data Replication Build number, date, and protocol version.

Tip: You can also view this information in the Version section of the Property table located at the bottom of the **Server Manager** > **Servers** view.

System codepage

The language and corresponding code page number used on the system.

System name

The name of the operating system.

OS version

The version of the operating system kernel.

OS release

The release level of the operating system kernel.

Machine

The machine hardware name and processor type.

Node name

The host name for the machine that runs the Server Manager.

License expires in *number* days.

The number of days before the Data Replication license expires.

Available space for the Server Manager directories, MB

The amount of disk space, in megabytes, that is available for the following directories that the Server Manager uses:

- Logs directory, which contains Server Manager logs
- Configs directory, which contains replication configurations
- Intermediate files directory (output/*configuration_name*) , which contains intermediate files
- DBSYNC_HOME directory, which is the Data Replication root installation directory

5. When you are finished, close the **Properties** dialog box.

Related Topics:

- "Common Replication Problems" on page 381

Configuring the Server Manager for HTTPS Communication

To provide for secure communication over a network, Data Replication supports HTTPS connections with one-way authentication for Server Manager instances. You can use HTTPS connections for all or some Server Manager instances.

Tip: For testing purposes, configure the Server Manager to use both HTTP and HTTPS connections simultaneously. For more information, see "Editing Properties for the Main Server or a Subserver" on page 138.

1. Acquire a certificate signed by a certificate authority (CA) or use a self-signed certificate in PEM format to prepare the Server Manager instances to accept HTTPS connections.

   To generate a self-signed certificate, use the OpenSSL utility. Save the generated certificate under the name cert.pem in the *DataReplication_installation* directory on the system where you run the Server Manager instance.

   Note: For secure communication, generate a cert.pem file for each Server Manager instance.

For example, the following OpenSSL command generates a self-signed certificate in PEM format:

```
openssl req -x509 -days 365 -nodes -newkey rsa:1024 -keyout cert.pem -out cert.pem
```

When the OpenSSL utility prompts you to enter the common name for the certificate, enter the IP address or host name that you use to connect to the Server Manager instance from the Data Replication Console.

2. Enable HTTPS communication for the Server Manager by using one of the following methods:

   - In the Data Replication Console, on the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user. On the **Server Manager** tab > **Servers** view, right-click the server row and click **Properties**. In the **Properties** dialog box, click the **Advanced Settings** tab and set the **HTTP** property to 1. Then restart the Server Manager.
   - If you run the Server Manager in the foreground, restart it with the https command line parameter set to 1. Optionally, also include the https_port command line parameter to specify the port for HTTPS connections. For example:

     ```
     server_manager.exe https=1 https_port=8090
     ```

     Valid values are integers from 1 through 65535. By default, the Server Manager uses port 8089 for HTTPS connections.
     On Windows, if you run the Server Manager as a service, stop and remove the service and then install the Server Manager service again with the https and https_port parameters. Then restart the service. For example:

     ```
     server_manager.exe RUN_AS_SERVICE -i https=1
     server_manager.exe RUN_AS_SERVICE -s
     ```

     On Linux and UNIX, if you run the Server Manager as a daemon, restart the daemon with the https and https_port parameters. For example:

     ```
     ./server_manager.sh https=1
     ```

   Note: Regardless of which method you use, if you restart the Server Manager later, you do not have to enable HTTPS communication again.

To connect to the Server Manager for which you enabled HTTPS communication in the Data Replication Console, select the **Use TLS** option.

Related Topics:

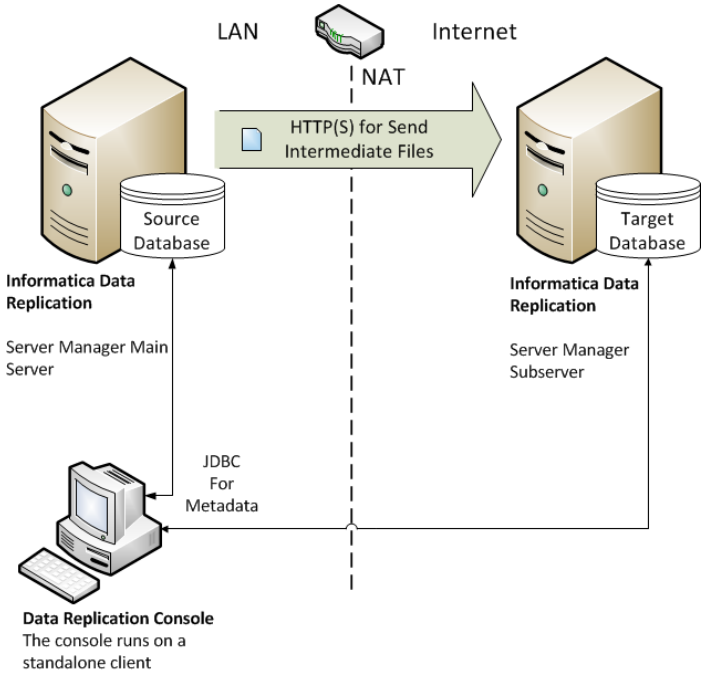
- "Connecting to the Server Manager Main Server" on page 128

Configuring the Server Manager Main Server to Run with NAT

Data Replication supports deployment topologies in which the Server Manager Main server is secured with Network Address Translation (NAT). NAT translates IP addresses for internal use in a local area network

(LAN) to IP addresses for external use. Often, the LAN is behind a firewall. Data Replication does not support NAT for Server Manager subservers.

The following image shows a topology in which the source database and the Server Manager Main server are deployed in a LAN and the target database and Server Manager subserver are deployed on a remote server:



When the Server Manager Main server sends data to the Server Manager subserver on the remote system, NAT converts the IP address that is used in the LAN to a public IP address.

Data Replication tries to use faster bidirectional connections by default. However, NAT prevents the Server Manager subserver from initiating a connection to the Server Manager Main server. Bidirectional connections work only if the Server Manager subservers can address the Server Manager Main server directly without NAT.

Alternatively, you can enable unidirectional connections that allow the Server Manager Main server in the LAN with NAT to communicate with subservers on remote servers. To do so, perform the following steps for the Server Manager Main server and each subserver in the replication configuration:

1. On the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user.
2. On the **Server Manager** tab > **Servers** view, select a server.
3. Click the **Properties** icon button on the Server Managers toolbar.
4. On the **Advanced Settings** tab, select **AsyncHttpEnabled** and click the **Edit** icon button.
5. Enter the value of 1 for **AsyncHttpEnabled** and click **OK**.
6. Click **Save**.
7. Restart the Server Manager for the change to take effect.

Related Topics:

- "Testing Connectivity Between Server Manager Instances" on page 134

Associating a Subserver with Another Main Server

The idradmin user can associate a Server Manager subserver with another Main server to move it to another replication group.

1. In the Data Replication Console, connect to the Main server of the replication group that includes the subserver you want to move. You must connect as the idradmin user.
2. On the **Server Manager** tab > **Servers** view, remove the subserver from the replication group.
3. Stop the subserver.
4. Delete all data from the subserver SQLite repository, SM.db3, by running the subserver executable with the clean option.

   For example, on Windows, use the following syntax:

   ```
   server_manager.exe clean
   ```
5. Restart the subserver.
6. In the Data Replication Console, connect to the Main server of the replication group to which you want to add the subserver. You must connect as the idradmin user.
7. On the **Server Manager** tab > **Servers** view, add the subserver to the replication group.

Deleting Subservers

In the Data Replication Console, the idradmin user can delete subservers from the Server Manager Main server.

Before you delete a subserver, perform the following prerequisite tasks for each replication configuration that is connected to the subserver that you want to delete:

- On the **Server Manager** tab > **Schedules** view, stop all running tasks. For more information, see "Stopping and Restarting a Scheduled Task Manually" on page 293.
- On the **Server Manager** tab > **Configs** view, change to another source or target subserver. For more information, see "Editing a Replication Configuration" on page 340.
- On the **Server Manager** tab > **Tasks** view, if you created user-defined tasks that run on the subserver that you want to delete, change the subserver on which user-defined tasks run. For more information, see "Editing Tasks" on page 291.

1. On the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user.
2. On the **Server Manager** tab > **Servers** view, select the subserver you want to delete.
3. Click the **Delete** icon button on the Server Managers toolbar, or right-click the subserver and click **Delete**.

   The Console displays a confirmation prompt.
4. Click **Yes**.

   The subserver is removed from the Server Managers list.

Related Topics:

- "Editing a Replication Configuration" on page 340
- "Editing Tasks" on page 291
- "Stopping and Restarting a Scheduled Task Manually" on page 293

CHAPTER 8

Creating and Managing User Accounts

This chapter includes the following topics:

User Account Overview

In the Data Replication Console, you can define multiple user accounts to control which users can connect to a Server Manager Main server and perform certain tasks in the Console.

For security purposes, each user must connect to the Main server under a user account. Multiple users can connect to a Main server at the same time. However, only one instance of a particular user can connect to the Main server at a time.

Users and Privileges

Data Replication provides two types of user accounts: the idradmin user account and regular replication user accounts. Initially, you must connect to the Server Manager as the idradmin user with a blank password. After you connect to the Server Manager Main server the first time, you can change the password for the idradmin user and create other user accounts.

Data Replication uses an ownership mechanism to determine user privileges. Ownership pertains to the following replication objects:

- Configurations

- Schedules
- Tasks
- Connections
- Environment variable lists
- Subscribers

When you create a replication object, you specify an owner for it.

Note: Object ownership completely determines user permissions. Neither the idradmin user nor a regular replication user can customize user permissions in the Data Replication Console.

The idradmin user has all rights and permissions. The idradmin user can modify all of the replication objects and start or stop all of the replication tasks and schedules. The idradmin user can also perform the following administrative tasks:

- Create and manage user accounts.
- Add subservers to a replication server group.
- Edit the Server Manager settings.
- View execution logs of replication tasks.
- View database connections.
- Manage open transactions.
- Deploy configurations.
- Manage database supplemental logging.
- Manage user notifications.

If you connect to a Main server as a regular replication user, you can perform only the following tasks:

- Modify only the replication objects for which you are designated as the owner.
- Start and stop the replication tasks and schedules for which you are designated as the owner.
- Run replication jobs only for the configurations for which you are designated as the owner.
- View execution logs of replication tasks for which you are designated as the owner.
- View database connections for the configurations for which you are designated as the owner.
- Manage open transactions for the configurations for which you are designated as the owner.
- Deploy the configurations for which you are designated as the owner.
- Manage database supplemental logging for the configurations for which you are designated as the owner.

Server Manager Security Policies

By default, Data Replication does not enforce any security policies. However, you can configure Server Manager security policies that enforce account lockout, password strength, and password expiration.

Account Lockout

You can configure the maximum number of times that users can retry entering a password for logging in to the Server Manager Main server from the Data Replication Console or the Server Manager Command-Line Interface (CLI). If a user fails to enter a correct password after the maximum number of retries, the Server Manager temporarily locks out the user account.

Use the Server Manager MaxLoginAttempts and LoginBlockTimeout advanced properties to specify the maximum number of invalid login attempts and the account lockout duration.

Note: The idradmin user can unlock an account. To unlock an account, click **File** > **Users** and then click **Unlock Account** in the **Users** dialog box.

Password Strength

The Data Replication Console and the Server Manager CLI can enforce the following password-strength policies when you create or edit a user account or connect to the Server Manager Main server:

- **Minimum password length**. Use the Server Manager CheckPasswordLength advanced property to specify the minimum password length.
- **Use of both uppercase and lowercase characters**. Use the Server Manager PasswordCaseSensitive advanced property to indicate whether passwords for user accounts must include both uppercase and lowercase characters.
- **Use of numbers**. Use the Server Manager PasswordContainNumbers advanced property to indicate whether passwords for user accounts must include one or more numbers.
- **Use of punctuation and symbols**. Use the Server Manager PasswordContainSpecialCharacters advanced property to indicate whether passwords for user accounts must include one or more special characters.

Password Expiration

Data Replication can enforce a password expiration policy for user accounts. Use the Server Manager ExpirationPeriodForPasswords advanced property to specify the number of days after which passwords for user accounts expire. By default, passwords never expire.

Creating a User Account

The idradmin user can create accounts for other users that allow the users to connect to the Server Manager Main server.

1. On the **Server Manager** tab > **Configs** view, connect to the Server Manager Main Server as the idradmin user.
2. Click **File** > **Users**.
3. In the **Users** dialog box, click the **New** toolbar button.
4. Enter the user login name and password, and then enter the password again to confirm it.

   Note: The user login name can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates login names that are longer than 50 characters.
5. Click **OK**.

   The user account row appears in the **Users** dialog box.

Changing the Password for Your User Account - Replication User

If the password for your account is unknown, contact the idradmin user.

Regular replication users can change the password that is used to connect to the Server Manager Main server for their own user accounts only.

1. On the **Server Manager** tab > **Configs** view, connect to the Server Manager Main server under your user account.
2. Click **File** > **Settings** to open the **Settings** dialog box.
3. On the **Password Changing** tab, enter the old password and a new password, and then enter the new password again to confirm it.
4. Click **Change**.
5. Click **Close**.

Changing the Password for a User Account - idradmin User

The idradmin user can change a password for any user account, including the idradmin user account.

1. On the **Server Manager** tab > **Configs** view, connect to the Server Manager Main server as the idradmin user.
2. Click **File** > **Users**.
3. In the **Users** dialog box, select a user account for which to change the password.
4. Click the **Edit** button on the toolbar, or right-click the user account row and click **Edit**.
5. In the **Editing** dialog box, enter the old password and a new password, and then enter the new password again to confirm it.
6. Click **Save**.

Unlocking a User Account

The idradmin user can unlock a user account. The Server Manager temporarily locks out a user account if the user fails to enter a correct password after the maximum number of retries.

1. On the **Server Manager** tab > **Configs** view, connect to the Server Manager Main server as the idradmin user.
2. Click **File** > **Users**.
3. In the **Users** dialog box, select the user account that you want to unlock.
4. Click the **Unlock** button on the toolbar or right-click the user account row and click **Unlock**.

Resetting the Password for the idradmin Account

If the password for the idradmin account is unknown, you must reset it to the default empty value before you can specify another password value.

1. Stop the Server Manager Main server.

   Stopping the Server Manager stops the running replication tasks and schedules.

2. Start the Server Manager Main server with the reset_idradmin command-line parameter. Use the following commands:

   - On Windows:

     ```
     server_manager.exe reset_idradmin
     ```

   - On Linux or UNIX:

     ```
     ./server_manager.sh reset_idradmin
     ```

   The Server Manager resets the password for the idradmin account to empty and then stops. Also, if the idradmin account was locked because you reached the maximum number of failed login attempts set by the MaxLoginAttempts property, the idradmin account is unlocked.

3. Restart the Server Manager Main server.

4. From the Data Replication Console, connect to the Server Manager Main server as the idradmin user using the empty password.

   If you want to change the password value from empty to a specific value, see "Changing the Password for a User Account - idradmin User" on page 163.

5. Restart all of the replication tasks and schedules.

CHAPTER 9

Creating and Managing Connections

This chapter includes the following topics:

Connections Overview

You can create source and target connections from the **Server Manager** tab > **Connections** view before you define any replication configurations that will use the connections. Alternatively, you can define source and target connections from the **Source Database** and **Target Database** tabs when you create a replication configuration.

This chapter focuses on creating and managing connections from the **Server Manager** tab.

Note: All database connection definitions are stored in the Server Manager SQLite database. When you assign a connection to a replication configuration, the connection information is also stored in the config SQLite database.

All connections that you create either from the **Connections** view or when you create a configuration are listed on the **Server Manager** tab > **Connections** view and have a unique name. You can assign a connection to multiple configurations.

If you need to edit a connection, you must do so from the **Server Manager** tab > **Connections** view. An updated connection will be used by all of the configurations to which it is assigned. Edit a connection and test it before you open any configuration that uses it. When you open a configuration in the Data Replication Console, the Console validates the connection information and reports an error if the connection is not valid.

Also, you can assign another existing connection definition to a configuration from the **Server Manager** tab > **Configs** view.

Related Topics:

- "Defining the Source Database" on page 176
- "Defining the Target Database" on page 187
- "Creating a Source or Target Connection from the Server Manager Tab" on page 166

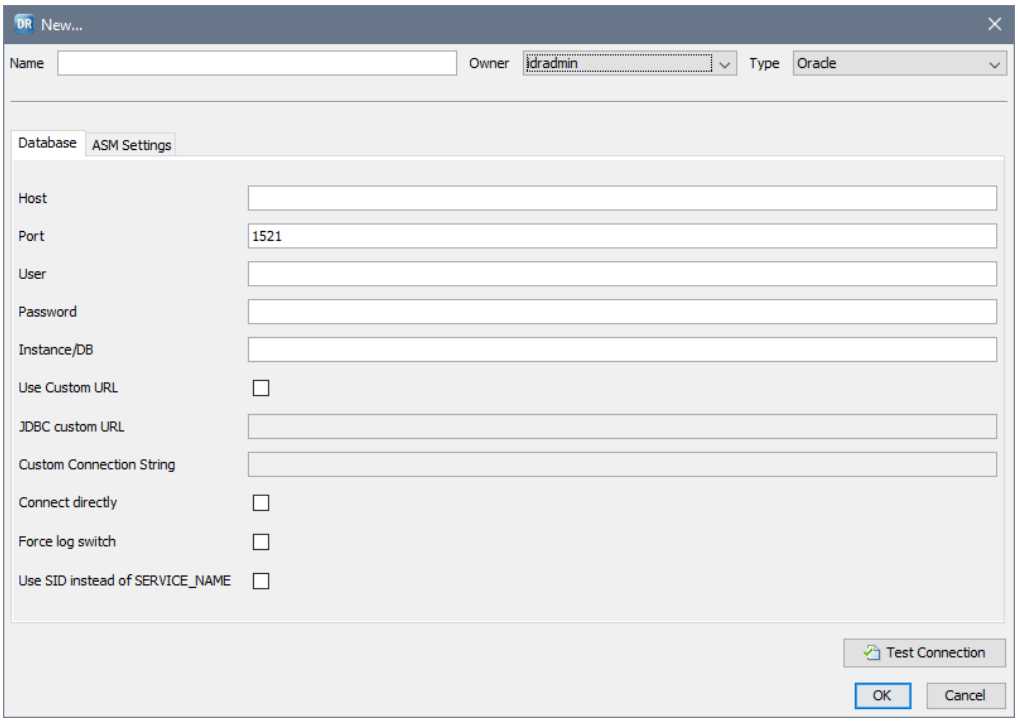
- "Editing a Source or Target Connection" on page 172
- "Assigning a Different Source or Target Connection to a Configuration" on page 173

Creating a Source or Target Connection from the Server Manager Tab

You can create a source or target connection under a unique name from the **Server Manager** tab > **Connections** view.

1. On the **Server Manager** tab > **Connections** view, click the **New** icon button on the Connections toolbar.

   The **New** dialog box appears.



   Note: For the Oracle connection type only, the **New** dialog box contains a **Database** tab and an **ASM Settings** tab.

2. Enter the database connection information.

The following table describes the fields for all source and target connection types:

Field	Source and Target Types	Description	Usage
Name	All	A unique name for the connection.	Required. Connection names can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates connection names that are longer than 120 characters.
Owner	All	The Data Replication user who is designated as the owner of the connection.	Required. The idradmin user can use any predefined connection.
Type	All	The source or target type. Options are: - Amazon Redshift - Apache Kafka - Cloudera - DB2 for LUW - Flat File - Greenplum - Hortonworks - Microsoft SQL Server - MySQL - Netezza - Oracle - PostgreSQL - Teradata - Vertica	Required. - For more information about the **Amazon Redshift** option, see "Defining Amazon Redshift Targets" on page 194. - For more information about the **Apache Kafka** option, see "Defining Apache Kafka Targets" on page 195. - For more information about the **Flat File** option, see "Defining Flat File Targets on a File System" on page 198. - For more information about the **Cloudera** and **Hortonworks** options, see "Defining Flat File Targets on a Hadoop Distributed File System" on page 198.
Version	- Cloudera - Hortonworks	The version of the selected target type.	Required.
Host	All, except for Apache Kafka and Flat File	The host name or IP address of the machine where the source or target runs.	Required.
Port	All, except for Apache Kafka and Flat File	The port number that Data Replication uses to connect to the source or target. Valid values are integers from 1 through 65535. The value that is displayed by default depends on the **Type** value.	Required.

Field	Source and Target Types	Description	Usage
User	All, except for Apache Kafka and Flat File	A user name that has the authority to write to the source or target.	Required.
Password	All, except for Apache Kafka and Flat File	A valid password for the specified user.	Required.
Path	- Cloudera - Hortonworks	The HDSF directory that contains the data files that Data Replication generates.	Required. The user ID that you specify for connecting to HDFS must have write permissions on this directory. **Note:** To browse HDFS, the Data Replication Console requires Java Runtime Environment (JRE) 1.7 or 1.8.
Instance/DB	- Amazon Redshift - DB2 for LUW - Greenplum - PostgreSQL - Netezza - Oracle - Vertica	An Oracle instance name or a database name.	Required if this field is displayed for the selected database type.

Field	Source and Target Types	Description	Usage
Use Custom URL	All, except for Apache Kafka, Cloudera and Hortonworks	Select this check box to provide custom connection strings for the JDBC or ODBC drivers. Use custom connection strings in the following situations: - You use a Data Source Name (DSN) with an ODBC driver to connect to the source or target. - You use a user name for connecting to the source or target from the Data Replication Console that is different from the user name under which the replication job runs. In this case, the user that connects to the source or target from the Data Replication Console must have privileges to read metadata. Also, the user that connects to the source or target when the replication job runs must have privileges to write DML changes to the target. - Multiple ODBC driver versions are installed on one Windows system. In this case, specify the name of the driver that you want to use in the **Custom connection string** field.	After you select this check box, the **JDBC custom URL** and **Custom Connection String** fields become available. For a DB2 source connection, select this check box if you need to specify an authentication method in a connection string.
JDBC custom URL	All, except for Apache Kafka, Cloudera and Hortonworks	A connection URL for the JDBC driver that the Data Replication Console uses. This URL has the following format: `jdbc:<subprotocol>:<subname>`	This field is available only if you selected **Use custom URL**. **Important:** JDBC custom connection strings always use the values that are specified in the **User** and **Password** fields. Ensure that any JDBC custom connection string that you enter does not include a user name and password. **Caution:** The Data Replication Console does not validate JDBC custom connection strings. Ensure that any JDBC custom connection string that you enter is valid for the current connection.

Field	Source and Target Types	Description	Usage
Custom Connection String	All, except for Apache Kafka, Cloudera and Hortonworks	A connection string for the ODBC driver that the Applier uses to access the source or target. If you use an ODBC Data Source Name (DSN) to connect to the source or target, provide the DSN connection string. On Windows, if you installed multiple ODBC driver versions on one system, use this setting to specify a name for the driver that you want to use.	This field is available only if you selected **Use Custom URL**. **Important:** Ensure that any custom connection string that you specify includes a user name that has the authority to write to the target database. A custom connection string can include the password for this user, or you can specify the password in the **Password** field. If you specify the password in both the custom connection string and **Password** field, Data Replication uses the password in the custom connection string. For Microsoft SQL Server Cluster sources and targets, provide the connection string in the following format: `DRIVER=SQL_Server_ODBC_driver;Server=virtual_IP;Network=DBMSSOCN;Trusted_Connection=Yes;Port=port;Address=virtual_IP,port;` For Oracle sources, if the Extractor fails to connect to the source by using the custom connection string, the Extractor tries to connect to the source again. For the second connection attempt, the Extractor builds the connection string based on the database connection or the connection information that you manually enter. To disable connection retries, set the global.fallback_connection runtime parameter to 0. For DB2 sources that use UTF-8 accented and non-Latin characters, if you run InitialSync on Linux or UNIX, add the IANAAppCodePage parameter to the connection

Field	Source and Target Types	Description	Usage
			string with a setting of 106. For example: `DRIVER={DB2};IpAddress=IP_address;port=port;UID=user;PWD=password;Database=db_name;IANAAppCodePage=106` For Oracle targets, if the Applier fails to connect to the target by using the custom connection string, the Applier tries to connect to the target again. For the second connection attempt, the Applier builds the connection string based on the database connection or the connection information that you manually enter. To disable connection retries, set the global.fallback_connection runtime parameter to 0. For Teradata targets, if you use a version of the Teradata ODBC driver that is earlier than 15.00.00.04 and include special characters in the user name or password for connecting to the target, you must specify a custom connection string. In the connection string, enclose the user ID or password that contains the special characters with double quotation marks. **Caution:** The Data Replication Console does not validate ODBC custom connection strings. Ensure that any ODBC custom connection string that you enter is valid for the current database connection.
Connect directly	Oracle	Reserved for future use.	-
Force log switch	Oracle	Select this check box if you capture data only from Oracle archived redo logs in a RAC environment.	Required for Oracle RACs. The Extractor issues the command to archive online redo logs prior to capturing the change data.

Field	Source and Target Types	Description	Usage
Use SID instead of SERVICE_NAME	Oracle	Select this check box to use the Oracle SID instead of the SERVICE_NAME to build the connection string.	Optional for Oracle connections.
Use SSL	- Amazon Redshift - Greenplum	Select this check box to use an HTTPS connection that is secured by the Secure Sockets Layer (SSL) to safely send SQL requests to Amazon Redshift or Greenplum targets.	If you clear this check box, Data Replication uses a connection that is not secured by SSL. Transmission of SQL requests to Amazon Redshift or Greenplum targets is insecure. **Important:** Data Replication always uses a secure HTTPS connection to load source data to the Amazon Simple Storage Service (Amazon S3).

Note: If the source or target is in an Oracle RAC, you can enter connection information for any running Oracle instance in the RAC.

3. Click **Test Connection** to verify the database connection.
4. If you use Oracle ASM, configure a connection to an Oracle ASM instance. Click the **ASM Settings** tab and then enter ASM connection information. For more information, see "Configuring a Connection to an Oracle ASM Instance" on page 182.
5. Click **OK** to save the connection under the specified name.

   In the **Connections** view, the new connection appears in the connections list under the appropriate database type.

Editing a Source or Target Connection

You can edit a source or target connection only from the **Server Manager** tab > **Connections** view. You cannot edit a connection from an open replication configuration. If you edit a connection that is assigned to one or more replication configurations, all of those configurations use the updated connection information.

1. On the **Server Manager** tab > **Connections** view, select the connection that you want to edit in the **Connections** list.
2. Click the **Edit** icon button on the Connections toolbar.

   The **Editing** dialog box appears.
3. Enter changes in one or more fields.

   For field descriptions, see "Creating a Source or Target Connection from the Server Manager Tab" on page 166.

   Caution: Do not change the source or target type in the **Type** list. Changing the database type for a connection might corrupt the configurations that use the connection.

4. Click **Test Connection** to verify the updated connection.

   If the test is successful, a message displays "Connection successfully established." Click **OK**.

   Caution: The Data Replication Console does not validate any connection strings that you enter in the **JDBC custom URL** and **Custom Connection String** fields. If you edit either of these fields, ensure that the value you enter is valid for the connection.

5. Click **Save**.

   A confirmation prompt informs you that the Server Manager will stop all replications that use the connection. Click one of the following options:

   - **Yes**. Saves the connection edits and stops any replication configurations that use the connection.
   - **No**. Does not save the connection updates and does not stop the associated configurations.

Assigning a Different Source or Target Connection to a Configuration

You can change the source or target connection that is assigned to a replication configuration only from the **Server Manager** tab > **Configs** view.

1. On the **Server Manager** tab > **Configs** view, select the replication configuration to which to assign a different source or target connection.
2. Right-click the configuration row and click **Properties**, or click the **Properties** icon button on the Replication Configurations toolbar.

   The **Editing** dialog box appears and displays the connections that are currently assigned to the configuration.

3. Under **Source Server Manager** or **Target Server Manager**, click in the **Connection** column and select another connection.

   A confirmation prompt appears.

4. Click **Yes**.

   The Data Replication Console updates the connection information and object IDs in the configuration and displays a log of operations in a separate dialog box.

5. When the operations are done, click **Close**.
6. Optionally, select another environment variables list in the **Runtime environment** list.
7. Click **Save**.
8. If the replication configuration is open in the Data Replication Console, double-click the configuration row on the **Server Manager** tab > **Configs** view to update the configuration information in the Data Replication Console.

CHAPTER 10

Creating Replication Configurations

This chapter includes the following topics:

Related Topics:

- "Replication from a Single Source to a Single Target" on page 22

Replication Configuration

Use the Data Replication Console to create configurations that the InitialSync, Extractor, and Applier tasks use to materialize targets and replicate change data from sources to targets.

A configuration includes the following types of information:

- Connection information for the source and target databases
- The source and target tables selected for change data replication or target materialization
- Mappings between source and target tables and columns
- Column filtering conditions
- The location of source transaction logs
- Locations of the InitialSync, Extractor, and Applier executables and Extractor output directory
- Various runtime and advanced configuration settings

Note: You can define connections to the source and target either from the **Server Manager** tab > **Connections** view or when you define the source or target for the configuration.

The Data Replication Console stores configurations as SQLite .db files on the system where the Server Manager Main server runs. Each configuration has an owner that is specified when the configuration is created.

Task Flow: Creating a Replication Configuration

Perform the following tasks to create a replication configuration from a single source to a single target:

1. Connect to the Server Manager Main server. See "Connecting to the Server Manager Main Server" on page 128.
2. Click the **New** toolbar button, or click **File** > **New**.
3. In the **New** dialog box, enter a configuration name, select the configuration owner, and select the source and target Server Manager servers and environment variables lists.
   Configuration names are not case-sensitive. They can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates configuration names that are longer than 100 characters.

   If you are a regular non-administrative user and select idradmin as the owner, you will not be able to edit the configuration later.
4. Define the source database. See "Defining the Source Database" on page 176.
5. Define the target database. See "Defining the Target Database" on page 187.
6. (Optional) Generate target tables. See "Generating Target Tables and Audit Log Tables" on page 198.
7. Map source and target tables. See "Mapping Source and Target Tables Based on Wildcard Expressions" on page 213.

8. (Optional) Customize column mappings. See "Customizing Column Mappings" on page 240.
9. Specify the database logs for extraction processing. See "Specifying the Database Logs from Which to Extract Data" on page 251.
10. (Optional) Configure runtime settings. See "Configuring Runtime Settings" on page 252.
11. (Optional) Configure message logging. See "Configuring Message Logging" on page 266.
12. Save the configuration file. See "Saving Replication Configurations to the Main Server Manager" on page 267.

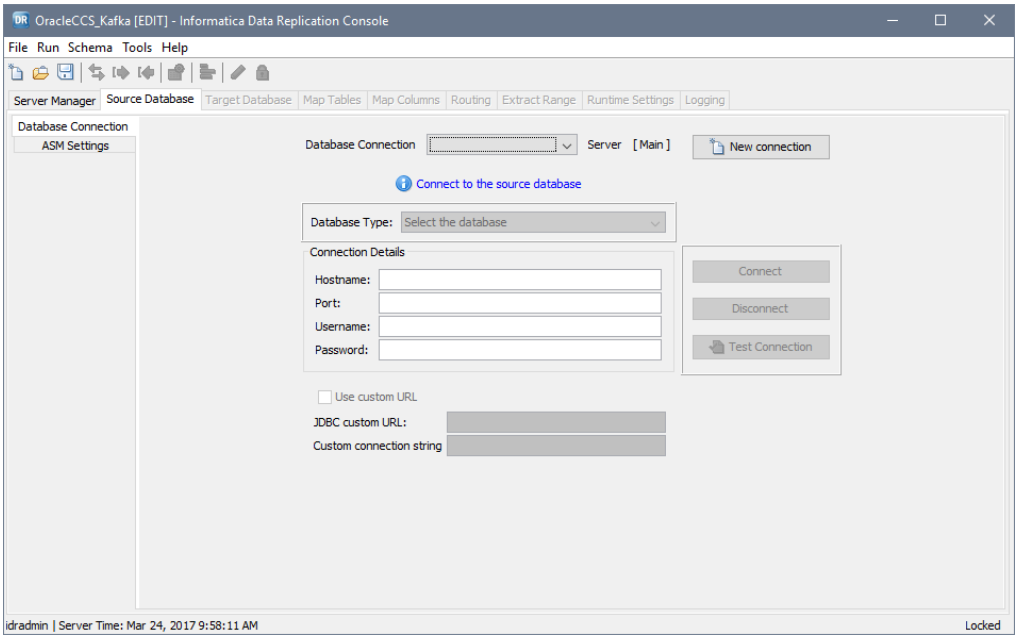
Defining the Source Database

For Data Replication to connect to the source database and extract data, you must select an existing source database connection or create a new connection. The connection information is saved to the configuration file and is required for all replication types.

1. Navigate to the **Source Database** tab > **Database Connection** view, if necessary.

   After you connect to the Server Manager Main server and enter a new configuration name, the **Source Database** tab > **Database Connection** view automatically appears.

   The following image shows the **Database Connection** view:



2. To use a predefined connection, select the connection from the **Database Connection** list.

   All of the listed connections are also listed on the **Server Manager** tab > **Connections** view. If you connected to the Server Manager as the idradmin user, you can use any predefined connection. If you are not the idradmin user, you must use a connection for which you are designated as the owner.

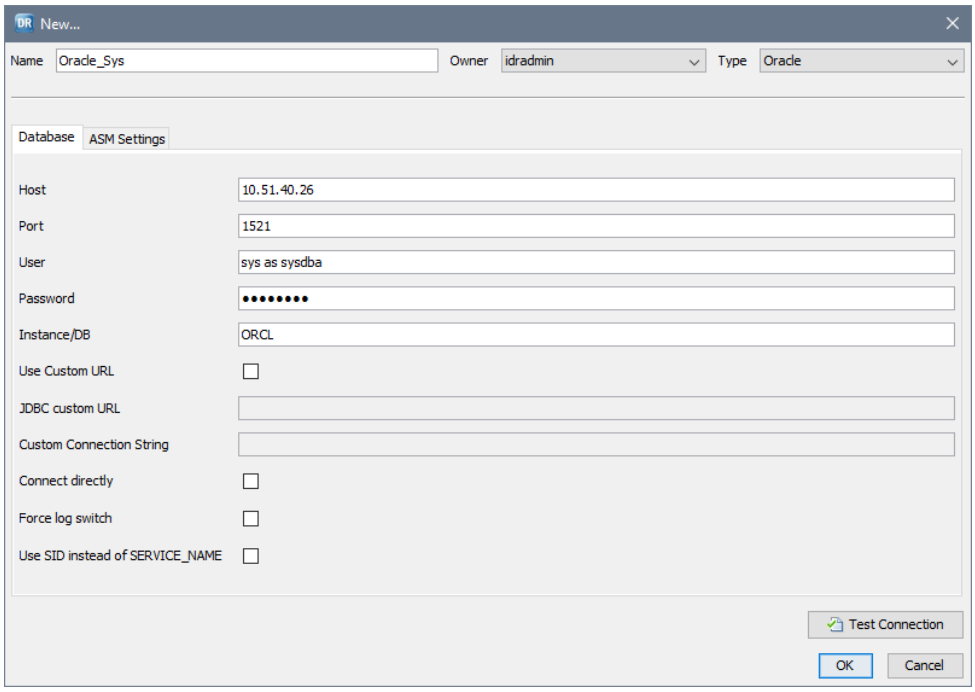
   The connection details are displayed in read only mode.

   Note: For more information about defining connections from the **Server Manager** tab > **Connections** view, see "Creating a Source or Target Connection from the Server Manager Tab" on page 166.

3. To create a new source database connection, perform the following steps:

   a. Click **New connection**. If you want to create a connection based on an existing connection, select a connection in the **Database Connection** list before you click **New connection**.

      The **New** dialog box > **Database** tab appears.



   b. To define a source database connection, enter connection information on the **Database** tab. The following table describes the fields on this tab:

Field Name	Sources	Description	Usage
Name	All	A name for the source database connection.	Required. Connection names can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates connection names that are longer than 120 characters.
Owner	All	The Data Replication user who is designated as the owner of the connection.	Required. The idradmin user can use any predefined connection.

Field Name	Sources	Description	Usage
Type	All	The source database type. Options are: - DB2 for LUW - Microsoft SQL Server - MySQL - Oracle	Required. **Warning:** After you save the configuration, do not change the source database type by editing the connection information from the **Server Manager** > **Connections** view or by editing configuration properties and switching to a connection for another database type from the **Server Manager** > **Configs** view. If you change the database type, replication processing will fail.
Host	All	Host name or IP address of the machine where the source database runs.	Required.
Port	All	Port number that Data Replication uses to connect to the source database. Valid values are integers from 1 through 65535.	Required.
User	All	A user name that has the authority to connect to the source database.	Required.
Password	All	A valid password for the specified user.	Required if you are manually entering connection details or if you are defining a custom URL for the JDBC connection. Optional if you are defining a custom URL for an ODBC connection.
Instance/DB	DB2, Oracle	A DB2 for Linux, UNIX, or Windows database name or an Oracle instance name that is defined in an ORACLE_SID environment variable.	Required for DB2 and Oracle sources.

Field Name	Sources	Description	Usage
Use Custom URL	All	Select this check box to provide custom connection strings for the JDBC or ODBC drivers. Use custom connection strings in the following situations: - You use a Data Source Name (DSN) with an ODBC driver to connect to the source database for data extraction. - The user that you use for connecting to the source database from the Data Replication Console is different from the user under which you run the replication job. In this case, the user that connects to the source from the Data Replication Console must have privileges that allow extraction of metadata. Also, the user that connects to the source when the replication job runs must have privileges that allow execution of SELECT queries. - On Windows, you installed multiple ODBC driver versions on one system. In this case, specify the name of the driver that you want to use in the **Custom Connection String** field.	After you select this check box, the **JDBC custom URL** and **Custom Connection String** fields become available. For a DB2 source, select this check box if you need to specify an authentication method in a connection string.
JDBC custom URL	All	A connection URL for the JDBC driver that the Data Replication Console uses. This URL has the following format: `jdbc:<subprotocol>:<subname>`	This field is available only if you selected **Use Custom URL**. **Important:** JDBC custom connection strings always use the values that are specified in the **User** and **Password** fields. Ensure that any JDBC custom connection string that you enter does not include a user name and password. **Caution:** The Data Replication Console does not validate JDBC custom connection strings. Ensure that any JDBC custom connection string that you enter is valid for the current database connection.

Field Name	Sources	Description	Usage

Custom Connection String	All	A connection string for the ODBC driver that the Extractor uses to access the database.
		If you use an ODBC Data Source Name (DSN) to connect to the source database, provide the DSN connection string.
		On Windows, if you installed multiple ODBC driver versions on one system, use this setting to specify a name for the driver that you want to use.
			This field is available only if you selected **Use Custom URL**.
			Important: Ensure that any custom connection string that you specify includes a user name that has the authority to read from the source database. A custom connection string can include the password for this user, or you can specify the password in the **Password** field. If you specify the password in both the connection string and **Password** field, Data Replication uses the password in the custom connection string.
			For Microsoft SQL Server Cluster sources, provide the connection string in the following format:
			`DRIVER=SQL_Server_ODBC_driver;Server=virtual_IP;Network=DBMSSOCN;Trusted_Connection=Yes;Port=port;Address=virtual_IP,port;`
			For Oracle sources, if the Extractor fails to connect to the source by using the custom connection string, the Extractor tries to connect to the source again. For the second connection attempt, the Extractor builds the connection string based on the database connection or the connection information that you manually enter. To disable connection retries, set the global.fallback_connection runtime parameter to 0.
			For DB2 sources that use UTF-8 accented and non-Latin characters, if you run InitialSync on Linux or UNIX, add the IANAAppCodePage parameter to the connection string with a setting of 106. For example:
			`DRIVER={DB2};IpAddress=IP_address;port=port;UID=user;PWD=password;Database=db_name;IANAAppCodePage=106`
			Caution: The Data Replication Console does not validate ODBC custom connection strings. Ensure that any ODBC custom connection string that you enter is valid for the current database connection.

Connect directly	Oracle	Reserved for future use.	-

Field Name	Sources	Description	Usage
Force log switch	Oracle	Select this check box if you capture data only from Oracle archived redo logs in a RAC environment.	Required for Oracle RACs. The Extractor issues the command to archive online redo logs prior to capturing the change data.
Use SID instead of SERVICE_NAME	Oracle	Select this check box to use the Oracle SID instead of the SERVICE_NAME to build the connection string.	Optional for Oracle sources.

Note: If the source is in an Oracle RAC, you can enter connection information for any running Oracle instance in the RAC.

c. Click **Test Connection** to verify the connection details.

d. If the source is in an Oracle ASM environment, click the **ASM Settings** tab and follow the steps in "Configuring a Connection to an Oracle ASM Instance" on page 182.

e. Click **OK**.

4. Click **Connect** to connect to the source.

   If the connection is successful, the **Target Database** tab is displayed and some other tabs become available. If the connection is not successful, Data Replication reports an error.

   Note: Saved connections are listed on the **Server Manager** > **Connections** view and are organized by database type.

Related Topics:

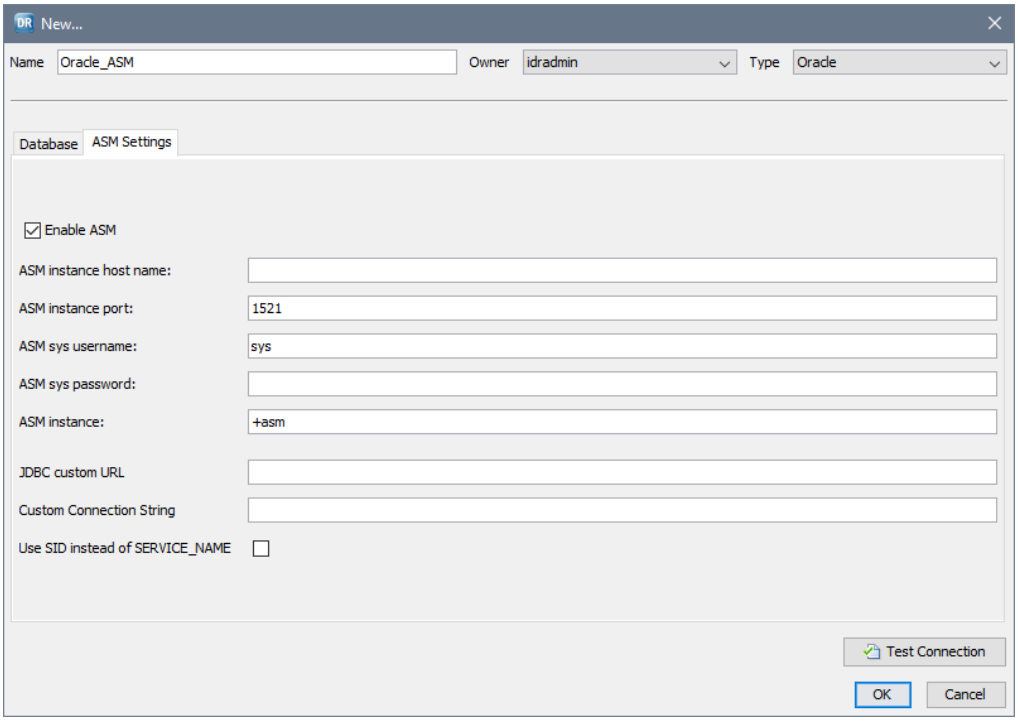
- "Configuring a Connection to an Oracle ASM Instance" on page 182
- "Defining the Target Database" on page 187
- "Connections Overview" on page 165

Configuring a Connection to an Oracle ASM Instance

If you use Oracle Automatic Storage Management (ASM) to manage storage for Oracle source data, you must enter connection settings for the ASM instance that you want to use for data replication.

1. In the **New** dialog box, select the **ASM Settings** tab.

   The following image shows this tab:



2. Select the **Enable ASM** check box.
3. Enter connection information for the ASM instance.

   Note: In an Oracle RAC with multiple ASM instances, you can enter connection information for any one of the running ASM instances.

   The following table describes the fields on the **ASM Settings** tab:

Field	Description
ASM instance host name	The host name or IP address of the system with the ASM instance.
ASM instance port	The port number that is used to connect to the ASM instance. Valid values are integers from 1 through 65535. The default value is 1521.
ASM sys username	The user name of the default Oracle user that was created at Oracle installation, or another user that you define. This user must have SYSDBA privileges.
ASM sys password	A clear text password for the ASM SYS user.

Field	Description
ASM instance	A service name for the ASM instance. The default value is +ASM. If you want to use the Oracle instance name instead, select the **Use SID instead of SERVICE_NAME** check box and then enter an Oracle instance name that is defined in an ORACLE_SID environment variable.
JDBC custom URL	A connection URL for the JDBC driver that the Data Replication Console uses. This URL has the following format: jdbc:*<subprotocol>*:*<subname>* **Important:** JDBC custom connection strings always use the values that are specified in the **ASM sys username** and **ASM sys password** fields. Ensure that any JDBC custom connection string that you enter does not include a user name and password.
Custom Connection String	A connection string for the ODBC driver that the Extractor uses to access the ASM instance. If you use an ODBC Data Source Name (DSN) to connect to the ASM instance, provide the DSN connection string. **Important:** Ensure that any custom connection string that you specify includes a user name that has the authority to read from the source database. A custom connection string can include the password for this user, or you can specify the password in the **ASM sys Password** field. If you specify the password in both the connection string and **ASM sys Password** fields, Data Replication uses the password in the custom connection string. On Windows, if you installed multiple ODBC driver versions on one system, use this setting to specify a name for the driver that you want to use.
Use SID instead of SERVICE_NAME	To use the Oracle instance name instead of the ASM instance name to establish connections to the source, select this check box. Then enter the Oracle instance name in the **ASM instance** field.

4. Click **Test Connection** to verify the ASM connection settings.
5. When you are done defining both the Oracle database connection and ASM connection, click **OK**.

Configuring Connections to Oracle RAC Sources for High Availability

For Oracle RAC sources that have multiple database and ASM instances, configure Data Replication support of failover to provide high availability of replicated data.

If you enable failover and a particular instance becomes unavailable for a Data Replication replication component, such as the Extractor or Data Replication Console, the replication component can connect to another instance.

For Data Replication to support failover of an Oracle RAC source, either use custom connection strings that provide connection information for all of the cluster instances or use a virtual IP address that is routed to an active cluster node to connect to the Oracle RAC.

Connecting to an Oracle RAC by Using Custom Connection Strings

For Data Replication to support failover in an Oracle RAC, you can define custom connection strings.

Custom connection strings provide connection information for all of the Oracle database instances and ASM instances in the RAC. If the Extractor or the Data Replication Console fails to connect to a particular database or ASM instance, it fails over to another available instance when reconnecting to the database or ASM.

You must provide OCI connection strings for the Extractor and JDBC connection strings for the Data Replication Console in one of the following ways:

- On the **Source Database** tab, click **New connection** to create a new source database connection. Then specify the connection strings in the **New** dialog box.
- On the **Server Manager** tab > **Connections** view, click the **New** icon button on the Connections toolbar. Then specify the connection strings in the **New** dialog box.
- On the **Server Manager** tab > **Connections** view, select an existing Oracle source database connection and click the **Edit** icon button on the Connections toolbar. Then specify the connection strings in the **Editing** dialog box.

The following steps include sample JDBC and OCI connection strings for a two-node RAC. When you customize the sample connection strings, replace the following variables with actual values:

- *username*. The Oracle user.
- *password*. A valid password for the specified user.
- *node1_IP*. The IP address or host name of the first Oracle instance.
- *node2_IP*. The IP address or host name of another Oracle instance.
- *port*. The Oracle listener port. Valid values are integers from 1 through 65535.
- *service_name*. The Oracle service name. Do not use the database instance name.
- *ASM_service_name*. The Oracle ASM service name. Do not use the ASM instance name.

Note: The sample connection strings specify two database instances or two ASM instances. If you have an Oracle RAC with more than two instances, add the ADDRESS parameter for each instance in the connection strings.

1. In the **New** or **Editing** dialog box for the connection, enter the following connection information for the Oracle database:
   a. In the **Type** list, select **Oracle**. The database type is already selected for an existing Oracle source database connection.
   b. Select the **Use Custom URL** check box.
   c. In the **JDBC custom URL** field, enter the JDBC connection string that includes connection information for all database instances.

      For example:

      ```
      jdbc:oracle:thin:@(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = node1_IP)
      (PORT = port))(ADDRESS = (PROTOCOL = TCP)(HOST = node2_IP)(PORT = port))
      (LOAD_BALANCE = off)(FAILOVER=on)(CONNECT_DATA = (SERVER = DEDICATED)
      (SERVICE_NAME = service_name)(FAILOVER_MODE =(TYPE = SELECT)(METHOD = preconnect)
      (RETRIES = 20)(DELAY = 3))))
      ```

   d. In the **Custom Connection String** field, enter the OCI connection string that includes connection information for all database instances.

      For example:

      ```
      username/password@(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = node1_IP)
      (PORT = port))(ADDRESS = (PROTOCOL = TCP)(HOST = node2_IP)(PORT = port))
      (LOAD_BALANCE = off)(FAILOVER=on)(CONNECT_DATA = (SERVER = DEDICATED)
      (SERVICE_NAME = service_name)(FAILOVER_MODE =(TYPE = SELECT)(METHOD = preconnect)
      (RETRIES = 20)(DELAY = 3))))
      ```

   e. In the **User** and **Password** fields, enter a database user ID and password for JDBC driver use.

      The Data Replication Console requires the user name and password to connect to the Oracle database because the JDBC connection string does not include this information.

   f. In the **New** dialog box, click **OK**. In the **Editing** dialog box, click **Save** and then click **Yes** at the confirmation prompt.

2. On the **ASM Settings** tab, enter the following connection information for the Oracle ASM instances:
   a. Select the **Use custom URL** option.
   b. In the **JDBC custom URL** field, enter the JDBC connection string that includes connection information for all ASM instances.

      For example:

      ```
      jdbc:oracle:thin:@(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = node1_IP)
      (PORT = port))(ADDRESS = (PROTOCOL = TCP)(HOST = node2_IP)(PORT = port))
      (LOAD_BALANCE = off)(FAILOVER=on)(CONNECT_DATA = (SERVER = DEDICATED)
      (SERVICE_NAME = ASM_service_name)(UR=A) (FAILOVER_MODE =(TYPE = SELECT)(METHOD =
      preconnect)(RETRIES = 20)(DELAY = 3))))
      ```

   c. In the **Custom Connection String** field, enter the OCI connection string for the Oracle ASM instances.

      For example:

      ```
      (DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = node1_IP)(PORT = port))(ADDRESS
      = (PROTOCOL = TCP)(HOST = node2_IP)(PORT = port))(LOAD_BALANCE = off)
      (FAILOVER=on)(CONNECT_DATA = (SERVER = DEDICATED)(SERVICE_NAME =
      ASM_service_name)(UR=A) (FAILOVER_MODE =(TYPE = SELECT)(METHOD = preconnect)
      (RETRIES = 20)(DELAY = 3))))
      ```

      Note: The OCI connection string for Oracle ASM instances does not require a user ID and password.
   d. In the **ASM sys username** and **ASM sys password** fields, enter an ASM user name and password for JDBC driver use.

      The Data Replication Console requires this user name and password to connect to the Oracle ASM instances because the JDBC connection string does not include this information.
   e. In the **New** dialog box, click **OK**. In the **Editing** dialog box, click **Save** and then click **Yes** at the confirmation prompt.

Connecting to an Oracle RAC by Using a Virtual IP Address or Host Name

If an Oracle RAC source has a virtual IP address or host name that supports failover and is routed to an active node, you can configure Data Replication to use this virtual IP address or host name.

If the Extractor or Data Replication Console fails to connect to a particular database instance or ASM instance, the virtual IP address or host name transparently fails over to another available instance.

You must specify the virtual IP address or host name and the global Oracle service name to connect to the Oracle database. You must also specify the virtual IP address or host name and the global ASM service name to connect to the Oracle ASM instance. Enter these connection details in one of the following ways:

- On the **Source Database** tab, click **New connection** to create a new source database connection. Then specify the connection strings in the **New** dialog box.
- On the **Server Manager** tab > **Connections** view, click the **New** icon button on the Connections toolbar. Then specify the connection strings in the **New** dialog box.
- On the **Server Manager** tab > **Connections** view, select an existing Oracle source database connection and click the **Edit** icon button on the Connections toolbar. Then specify the connection strings in the **Editing** dialog box.

1. On the **Database** tab of the **New** dialog box or **Editing** dialog box, enter the following connection information for the Oracle database:
   a. In the **Type** list, select **Oracle**.
   b. In the **Host** field, enter the virtual IP address or host name of an active Oracle database instance.

c. In the **Port** field, enter the Oracle listener port number. Valid values are integers from 1 through 65535.

d. In the **User** and **Password** fields, enter a database user ID and password that permits connection to the Oracle database.

e. In the **Instance/DB** field, enter the global Oracle service name. Do not use the Oracle instance name.

f. In the **New** dialog box, click **OK**. In the **Editing** dialog box, click **Save** and then click **Yes** at the confirmation prompt.

2. On the **ASM Settings** tab of the **New** dialog box or **Editing** dialog box, enter the following connection information for the Oracle ASM instance:

   a. In the **ASM instance host name** field, enter the virtual IP address or host name of an active Oracle ASM instance.

   b. In the **ASM instance port** field, enter the port number to use for connecting to the ASM instance. Valid values are integers from 1 through 65535.

   c. In the **ASM sys username** and **ASM sys password** fields, enter a user name and password that permits connection the Oracle ASM instance.

      This user must have SYSDBA privileges.

   d. In the **ASM instance** field, enter the global Oracle ASM service name. Do not use the ASM instance name.

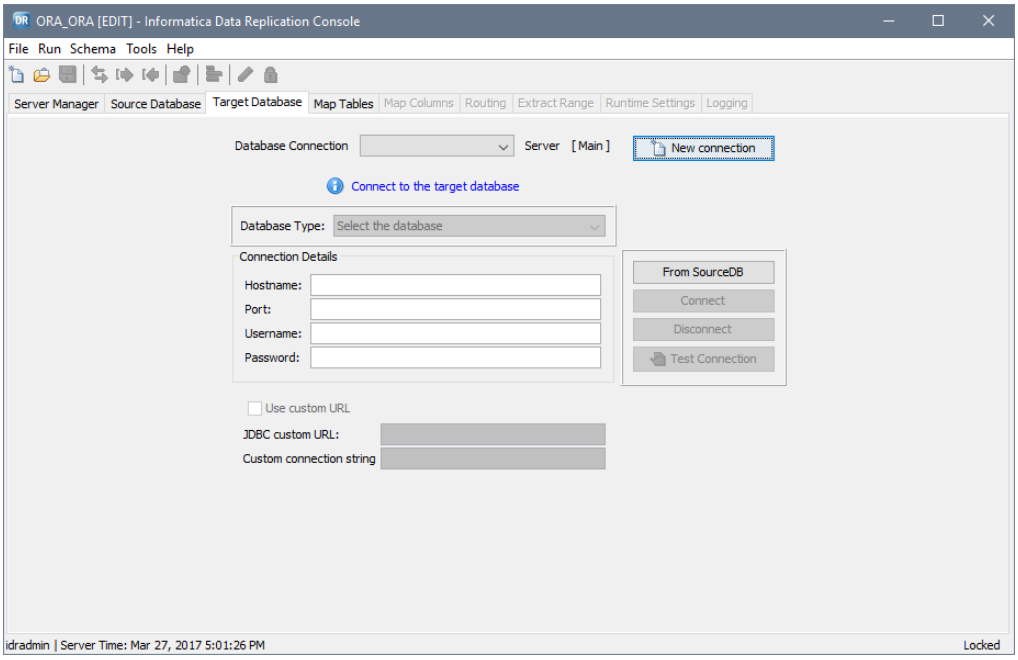
   e. In the **New** dialog box, click **OK**. In the **Editing** dialog box, click **Save** and then click **Yes** at the confirmation prompt.

Defining the Target Database

For Data Replication to connect to the target database and apply data, you must define a connection. You can select a connection that is already defined or create a new connection. You can create a connection from scratch, based on the source database connection, or based on another connection that you select.

1. Click the **Target Database** tab.

   The following image shows this tab:



2. To use a predefined connection, select the connection from the **Database Connection** list.

   All of the listed connections are also listed on the **Server Manager** tab > **Connections** view. If you connected to the Server Manager as the idradmin user, you can use any predefined connection. If you are not the idradmin user, you must use a connection for which you are designated as the owner.

   The connection details are displayed in read only mode.

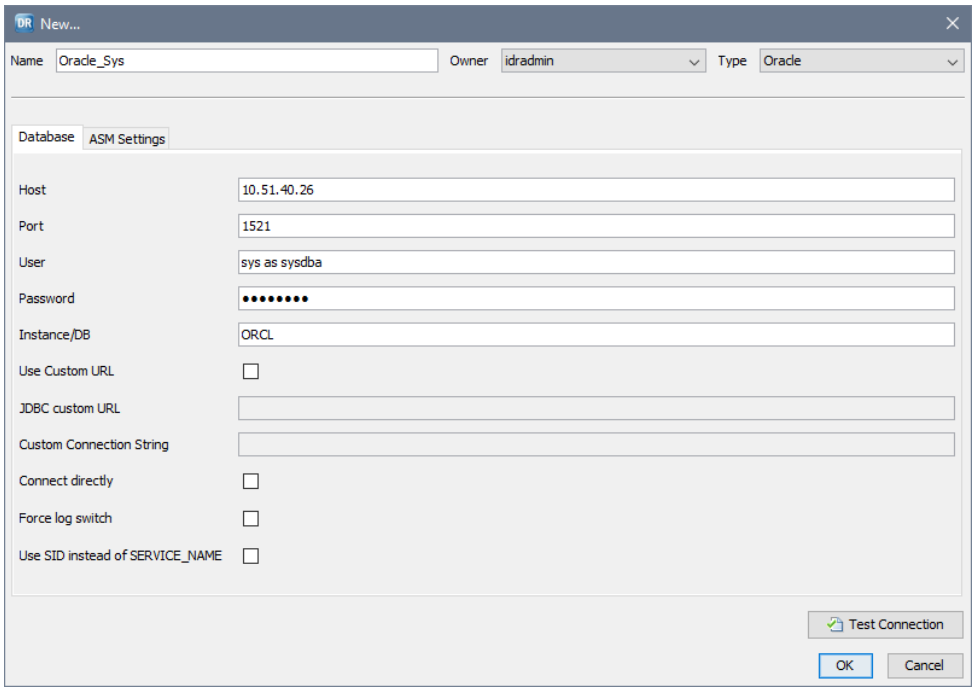
3. To use the source connection settings when the target tables are on the source system, click the **From SourceDB** button.

   The connection details are displayed in read only mode.

4. To create a new target connection, perform the following steps:

   a. Click **New connection**. If you want to create a connection based on an existing connection, select a connection in the **Database Connection** list before you click **New connection**.

      The **New** dialog box > **Database** tab appears.



   b. To define the target connection, enter connection information on the **Database** tab. The following table describes the fields on this tab:

Field	Target Types	Description	Usage
Name	All	A name for the target connection.	Connection names can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates connection names that are longer than 120 characters.
Owner	All	The Data Replication user who is designated as the owner of the connection.	Required. The idradmin user can use any predefined connection.

Field	Target Types	Description	Usage
Type	All	Select the target database type. Options are: - Amazon Redshift - Apache Kafka - Cloudera - DB2 for LUW - Flat File - Greenplum - Hortonworks - Microsoft SQL Server - MySQL - Netezza - Oracle - PostgreSQL - Teradata - Vertica	Required. - For more information about the **Amazon Redshift** option, see "Defining Amazon Redshift Targets" on page 194. - For more information about the **Apache Kafka** option, see "Defining Apache Kafka Targets" on page 195. - For more information about the **Flat File** option, see "Defining Flat File Targets on a File System" on page 198. - For more information about the **Cloudera** and **Hortonworks** options, see "Defining Flat File Targets on a Hadoop Distributed File System" on page 198. **Warning:** After you save the configuration, do not change the target database type by editing the connection information from the **Server Manager** > **Connections** view or by editing configuration properties and switching to a connection for another database type from the **Server Manager** > **Configs** view. If you change the database type, replication processing will fail.
Version	- Cloudera - Hortonworks	The version of the selected target type.	Required.
Host	All, except for Apache Kafka and Flat File	Host name or IP address of the machine where the target database runs.	Required.
Port	All, except for Apache Kafka and Flat File	Port number that Data Replication uses to connect to the target database. Valid values are integers from 1 through 65535.	Required.

Field	Target Types	Description	Usage
User	All, except for Apache Kafka and Flat File	A user name that has the authority to write to the target database.	Required.
Password	All, except for Apache Kafka and Flat File	A valid password for the specified user.	Required.
Path	- Cloudera - Hortonworks	The HDFS directory for the data files that Data Replication generates for the target.	Required. The user ID that you specified for connecting to HDFS must have write permissions on this directory. **Note:** To browse HDFS, the Data Replication Console requires Java Runtime Environment (JRE) 1.7 or 1.8.
Instance/DB	- Amazon Redshift - DB2 for LUW - Greenplum - Netezza - Oracle - PostgreSQL - Vertica	An Oracle instance name or a database name.	Required if this field is displayed for the selected database type.

Field	Target Types	Description	Usage
Use Custom URL	All, except for Apache Kafka, Cloudera and Hortonworks	Select this check box to provide custom connection strings for the JDBC or ODBC drivers. Use custom connection strings in the following situations: - You use a Data Source Name (DSN) with an ODBC driver to connect to the target database. - You use a user name for connecting to the target database from the Data Replication Console that is different from the user name under which the replication job runs. The user that connects to the target from the Data Replication Console must have privileges to read metadata. The user that connects to the target when the replication job runs must have privileges to write DML changes to the target. - Multiple ODBC driver versions are installed on one Windows system. In this case, specify the name of the driver that you want to use in the **Custom connection string** field.	After you select this check box, the **JDBC custom URL** and **Custom Connection String** fields become available.
JDBC custom URL	All, except for Apache Kafka, Cloudera and Hortonworks	A connection URL for the JDBC driver that the Data Replication Console uses. This URL has the following format: `jdbc:<subprotocol>:<subname>`	This field is available only if you selected **Use Custom URL**. **Important:** JDBC custom connection strings always use the values that are specified in the **User** and **Password** fields. Ensure that any JDBC custom connection string that you enter does not include a user name and password. **Caution:** The Data Replication Console does not validate JDBC custom connection strings. Ensure that any JDBC custom connection string that you enter is valid for the current database connection.

Field	Target Types	Description	Usage
Custom Connection String	All, except for Apache Kafka, Cloudera and Hortonworks	A connection string for the ODBC driver that the Applier uses to access the database. If you use an ODBC Data Source Name (DSN) to connect to the target, provide the DSN connection string.	This field is available only if you selected **Use Custom URL**. **Important:** Ensure that any custom connection string that you specify includes a user name that has the authority to write to the target database. A custom connection string can include the password for this user, or you can specify the password in the **Password** field. If you specify the password in both the custom connection string and **Password** field, Data Replication uses the password in the custom connection string. For Microsoft SQL Server Cluster targets, provide the connection string in the following format: `DRIVER=SQL_Server_ODBC_driver;Server=virtual_IP;Network=DBMSSOCN;Trusted_Connection=Yes;Port=port;Address=virtual_IP,port;` For Oracle targets, if the Applier fails to connect to the target by using the custom connection string, the Applier tries to connect to the target again. For the second connection attempt, the Applier builds the connection string based on the database connection or the connection information that you manually enter. To disable connection retries, set the global.fallback_connection runtime parameter to 0. For Teradata targets, if you use a version of the Teradata ODBC driver that is earlier than 15.00.00.04 and include special characters in the user

Field	Target Types	Description	Usage
			name or password for connecting to the target, you must specify a custom connection string. In the connection string, enclose the user ID or password that contains the special characters with double quotation marks. **Caution:** The Data Replication Console does not validate ODBC custom connection strings. Ensure that any ODBC custom connection string that you enter is valid for the current database connection.
Connect directly	Oracle	Reserved for future use.	-
Force log switch	Oracle	Select this check box if you capture data only from Oracle archived redo logs in a RAC environment.	Required for Oracle RACs. The Extractor issues the command to archive online redo logs prior to capturing the change data.
Use SID instead of SERVICE_NAME	Oracle	Select this check box to use the Oracle SID instead of the SERVICE_NAME to build the connection string.	Optional for Oracle targets.
Use SSL	- Amazon Redshift - Greenplum	Select this check box to use an HTTPS connection that is secured by the Secure Sockets Layer (SSL) to safely send SQL requests to Amazon Redshift or Greenplum targets.	If you clear this check box, Data Replication uses a connection that is not secured by SSL. Transmission of SQL requests to Amazon Redshift or Greenplum targets is insecure. **Important:** Data Replication always uses a secure HTTPS connection to load source data to the Amazon Simple Storage Service (Amazon S3).

Note: If the target is in an Oracle RAC, you can enter connection information for any running Oracle instance in the RAC.

c. If the target is an Oracle ASM instance, navigate to the **ASM Settings** tab and follow the steps in "Configuring a Connection to an Oracle ASM Instance" on page 182

d. Click **Test Connection** to verify the connection details.

e. Click **OK**.

5. If you have not already done so, click **Test Connection** to verify the connection details. Then click **OK**.

6. Click **Connect** to connect to the target.

   If you connect to the target successfully, the **Map Tables** tab opens. Otherwise, Data Replication reports an error.

Related Topics:

- "Defining the Source Database" on page 176
- "Connections Overview" on page 165

Defining Amazon Redshift Targets

To replicate change data to an Amazon Redshift target, define the target connection information and runtime settings as part of the configuration.

1. On the **Target Database** tab, click **New Connection**. If you want to create a connection based on an existing connection, select a connection in the **Database Connection** list before you click **New connection**.

   The **New** dialog box appears.
2. In the **Name** field, enter a name for the new Amazon Redshift connection.

   Note: Connection names can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates connection names that are longer than 120 characters.
3. Select an owner for the connection in the **Owner** list.
4. Select **Amazon Redshift** in the **Type** list.
5. Enter the host, port number, database, user ID, and password to use for connecting to the Amazon Redshift database.
6. On the **Runtime Settings** tab, enter Amazon S3 information in the following advanced runtime parameters:
   - redshift.s3.file_size
   - redshift.s3.bucket_name
   - redshift.s3.key_id
   - redshift.s3.path
   - redshift.s3.secret_key

Related Topics:

- "Data Replication Runtime Parameters" on page 396

Defining Apache Kafka Targets

To replicate change data to an Apache Kafka target, define target information such as the Kafka topic or topics to which messages will be published, the message format and encoding, and the Avro schema format. You must also specify the Kafka producer properties file that includes connection information.

Note: For Kafka targets, initial synchronization is not necessary.

1. On the **Target Database** tab, click **New Connection**. If you want to create a connection based on an existing connection, select a connection in the **Database Connection** list before you click **New connection**.

   The **New** dialog box > **Database** view appears.

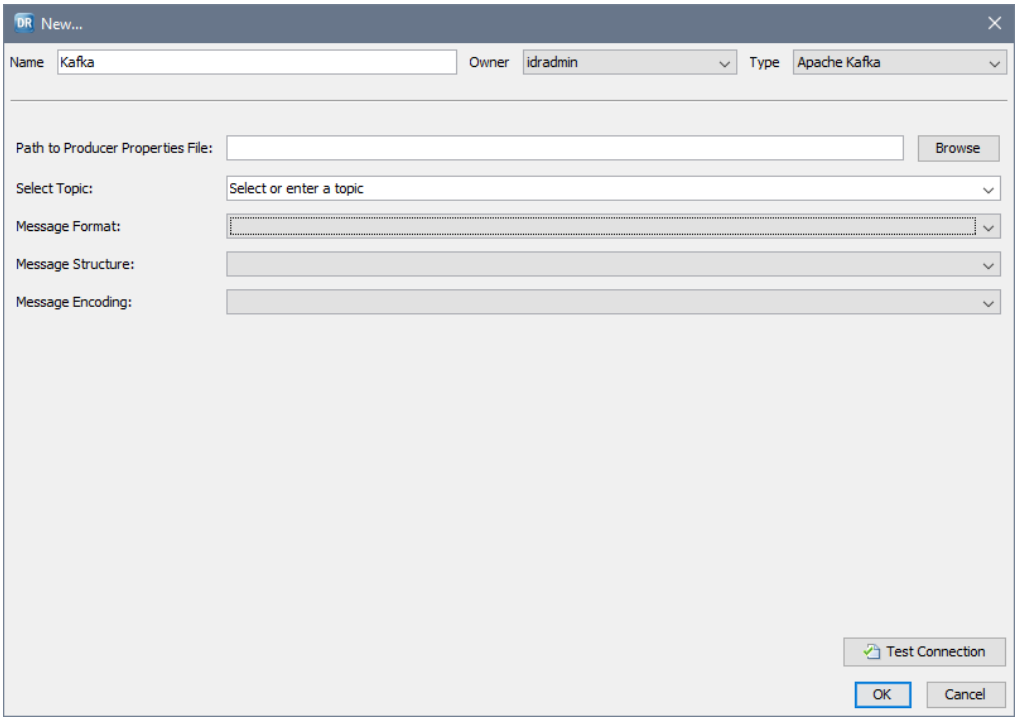
2. In the **Name** field, enter a name for the new Kafka connection.

   Note: Connection names can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates connection names that are longer than 120 characters.

3. In the **Owner** list, select an owner for the target definition.

4. In the **Type** list, select **Apache Kafka**.

   The following image shows the **New** dialog box for a Kafka target:



5. To the right of the **Path to Producer Properties File** field, click **Browse** to browse to the Kafka producer.properties file that Data Replication will use to communicate with the Kafka target.

   Note: You must have previously defined the file on the target Server Manager where the Applier runs for Data Replication to access the file. For more information about producer.properties files, see the Apache Kafka documentation.

6. In the **Select Topic** field, select one of the following methods for selecting the Kafka topics to which Data Replication writes messages that contain the source data:
   - Select **USE_TABLE_NAME** to direct Data Replication to use the Kafka topic names that match the mapped source table names, for each configuration that uses this connection.

     Note: By default, the auto.create.topics.enable parameter in the Kafka server.properties file is set to true. With this setting, Kafka automatically generates the topic or topics if they do not already exist when messages are sent to the topics. If you set the auto.create.topics.enable parameter to false, Kafka does not generate the topics. You must manually create the topics in Kafka before starting replication based on any configuration that uses this connection.
   - Enter the name of a single Kafka topic to which Data Replication will send the change data from all of the mapped source tables, for each configuration that uses this connection.

   Note: To append additional information from source tables to Kafka topic names, set either of the following advanced runtime parameters on the **Runtime Settings** tab > **Advanced Settings** view after you finish defining the Kafka target:
   - To include source table schema names in Kafka topic names, set the value of the apply.kafka.kafka_table_name_include_schema parameter to 1.
   - To specify a prefix for the Kafka topic names, enter the prefix in the apply.kafka.kafka_table_name_prefix parameter.
7. In the **Message Format** field, select **avro**.
8. In the **Message Structure** field, select the Avro format that Data Replication will use for the messages it sends to target topics. Options are:
   - **avroFlatSchemaFormatV1**. Formats messages in a flat Avro schema format, which lists all Avro fields in one record.
   - **avroNestedSchemaFormatV1**. Formats messages in a nested Avro schema format, which organizes each type of information in a separate record.
9. In the **Message Encoding** field, select the encoding method for Avro messages. Options are:
   - **binary**
   - **json**

   Tip: If Informatica Intelligent Streaming will consume the data written to the Kafka messaging system, use the binary encoding type. Informatica Intelligent Streaming does not support Kafka messages that use the JSON encoding type.
10. If you click **Test Connection**, Data Replication always reports that the test is successful, even though it cannot validate connectivity to a Kafka target. If the connection information is not valid, Data Replication will report a connection error when it tries to connect to the Kafka target.
11. Click **Connect** to connect to the Kafka target.

    If you connect to the target successfully, the **Map Tables** tab opens. Otherwise, Data Replication reports an error.

You can set many advanced runtime parameters for Kafka targets on the **Runtime Settings** tab > **Advanced Settings** view. If the default parameter values are not suitable for your environment, edit them. For example, you can customize the checkpoint file name and directory in the apply.kafka.kafka_checkpoint_file_name and apply.kafka.kafka_checkpoint_file_directory parameters.

Related Topics:

- "Data Replication Runtime Parameters" on page 396

Flat File Targets

You can configure Data Replication to write transactional changes to flat files instead of to a target database. Data Replication can generate these flat files on a file system or Hadoop Distributed File System (HDFS).

For flat file targets, you do not need to perform initial synchronization of the target files with InitialSync or Fast Clone before starting replication.

Flat File Generation and Format

The flat files to which Data Replication writes change data use a format that is similar to the audit log table schema. To store the generated flat files, Data Replication creates a directory hierarchy that groups the flat files by apply cycle.

During each apply cycle, the Applier creates a cycle_*cycle_sequence_number*.work subdirectory in the flat-file directory that you specified when defining the flat-file target. In this subdirectory, the Applier creates the following files:

- The schema.ini file that describes the schema and some settings for all of the target flat files.
- A text (.txt) file for each mapped source table, which contains the extracted change data. The Applier names these .txt files based on the name of the source table.

The schema.ini file contains a section for each replicated source table. Each section lists a sequence of columns for the rows in the corresponding flat file. The following table describes the columns in the schema.ini file:

Column	Description
OP_TIME	The date and time of an operation
OP_CODE	The code for the operation type
OP_CMT_SCN	The commit SCN
OP_CMT_TIME	The commit time of the operation
OP_XID	The transaction ID
OP_NUM_IN_TX	The sequence number of an operation in a transaction
<*source_column_name*>_OLD	The before value of the source column
<*source_column_name*>_NEW	The after value of the source column

The schema file also includes the following settings:

- **ColNameHeader**. Indicates whether the change data files include column headers. The Applier generates change data files without column headers.
- **Format**. Describes the format of the change data files. The Applier uses a semicolon (;) to delimit column values.

- **CharacterSet**. Specifies the character set that is used for change data files. The Applier generates change data files in the ANSI character set.

Important: Do not edit the schema.ini file.

Each target flat file stores change data from a single source table and is named based on the source table name. For each operation that occurred on the source table, Data Replication adds a record to the flat file, which is similar to a row in the audit log table.

Defining Flat File Targets on a File System

To define a file system target for generated flat files, use the **Flat File** option.

1. On the **Target Database** tab, select **Flat File** in the **Database Type** field.
2. Click the **Browse** button to specify a directory location for the generated flat files.

Defining Flat File Targets on a Hadoop Distributed File System

To define a Hadoop Distributed File System (HDFS) target for generated flat files, select **Cloudera** or **Hortonworks** as the database type.

1. On the **Target Database** tab, select **Cloudera** or **Hortonworks** in the **Database Type** list.
2. Enter the version, host, port number, and user ID to use for connecting to HDFS.
3. To specify the HDFS directory for the generated flat files, either click **Select** to browse to the directory or enter the path to the directory in the **Path** field.

   The user ID that you specified for connecting to HDFS must have write permissions on this directory.

   Note: To browse HDFS, the Data Replication Console requires Java Runtime Environment (JRE) 1.7 or 1.8.

Generating Target Tables and Audit Log Tables

For data replication jobs, the target table structure must be compatible with the source table structure.

The source and target columns must have compatible datatypes. Also, Informatica recommends that the names of source tables and columns match the names of the target tables and columns.

In the Data Replication Console, you can perform the following tasks:

- Generate target tables and audit log tables based on the source or target table schemas.
- Generate SQL CREATE TABLE scripts that you can run later to create target tables and audit log tables based on the source or target table schemas.

If the source and target database types are different, Data Replication uses the rules in the <ConversionRules> section of the *DataReplication_installation*\uiconf\DataTypes.xml file to convert source column datatypes to compatible target column datatypes.

Note: Data Replication does not generate target tables or audit log tables for Apache Kafka, Cloudera, Flat File, and Hortonworks targets.

Related Topics:

- "Datatype Conversion Rules" on page 52

Generating Target Tables and Audit Log Tables Based on Source Table Schema

You can generate target tables and audit log tables based on source table schema.

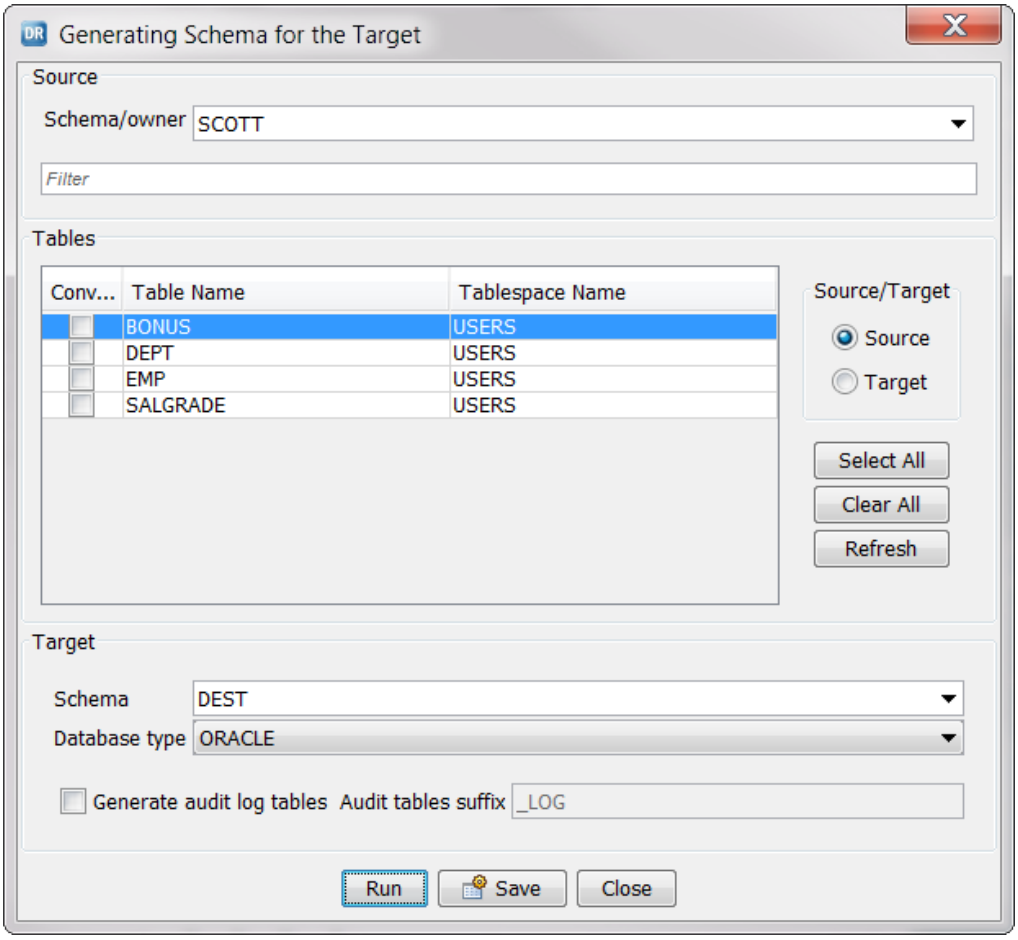
Before you begin, open the configuration and connect to the source and target databases.

1. Click **Schema** > **Generate Schema for the Target** on the menu bar, or click the **Generate Schema for the Target** icon button on the Replication Configurations toolbar.

   The **Generating Schema for the Target** dialog box appears.
2. Under **Source/Target**, verify that the **Source** option is selected.
3. In the **Schema/owner** field, select the schema or owner name for the source tables that you want to use to generate the target tables.

   The tables that match the selected schema or owner name are listed under **Tables**.

   The following image shows an example list of tables that match a schema or owner name of SCOTT:



4. To filter the list of tables, in the *Filter* field, enter the first few letters of the source table names that you want to use for generating target tables. For case-sensitive filtering, enclose the filter value in double quotation marks.

Only the tables that match the filter criteria are listed under **Tables**.

Tip: At any time, you can click **Refresh** to refresh the schemas for the source and target databases.

5. Select the tables to use for generating the target tables:
   - To select all listed tables, click **Select All**.
   - To select tables individually, select the **Convert** check box in the table row.
6. Under **Target**, in the **Database type** list, select the target database type.
7. In the **Schema** field, select a schema name for the target tables if the target database type is the one to which you are already connected and that is specified in the configuration. If you selected a different target database type in the **Database type** list, enter the correct schema name for the target tables manually.

   For Microsoft SQL Server targets, in the **Owner** field, select an owner for the target tables if you are already connected to the SQL Server target database. If you are connected to another type of target database but want to generate target tables for a SQL Server target, enter the owner name to use for the SQL Server target tables manually.
8. If you use Audit Apply or Merge Apply mode and need to generate audit log tables on the target, select the **Generate audit log tables** check box. In the adjacent **Audit tables suffix** field, enter a suffix for the generated audit log table names.
   - For Merge Apply, enter the suffix that is specified for the **Log table suffix for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view.
   - For Audit Apply, you can enter any alphanumeric string as the suffix for the audit log table names. However, if you want to use the **Map All** or **Wildcard map** button to quickly map all of the audit log tables to the corresponding source tables, you must specify the suffix that is specified in the **Log table suffix for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view.

   The default suffix is _LOG.

   Important: For Merge Apply mode, if you create a target table manually outside of the Data Replication Console, you must generate the audit log table based on the target table schema.
9. To create the target tables, perform one of the following actions:
   - To generate a script with SQL CREATE TABLE statements, click **Save**. Then in the **Save** dialog box, save the generated script to a directory of your choice. You can run the script on the target system later to create the target tables.
   - To create the tables on the target immediately, click **Run**. You can perform this action only for the target type to which you are already connected.

     Note:

     - If you cancel the Run operation while it is in progress, any tables that were already created on the target remain on the target. You can find the SQL CREATE TABLE statements for these tables in the %DBSYNC_HOME%/logs/ui/se.log file.
10. Click **Close**.

Related Topics:

- "Audit Log Tables" on page 34
- "Audit Apply" on page 32

Schema Generation Parameters

Data Replication uses default parameter settings when generating target tables based on source table schema unless you adjust the defaults in the *DataReplication_installation*/uiconf/default.cfg file.

Note: If the default.cfg file does not exist, you can create the file in the uiconf directory and then add parameters to it. For more information, see "Default.cfg File" on page 391.

The following parameters are related to schema generation:

schema_converter_use_quotes_in_tables_names={true|false}

Indicates whether to surround table names with brackets when Data Replication generates SQL CREATE TABLE scripts.

When you enable this option, Data Replication surrounds table names with the following symbols:

- Square brackets [ ] for Microsoft SQL Server
- Apostrophes (') for MySQL
- Double quotation marks (") for other target databases

Note: For Teradata targets, Data Replication surrounds table names with double quotation marks, regardless of this parameter setting.

The default value is true.

schema_converter_use_quotes_in_columns_names={true|false}

Indicates whether to surround column names with brackets when Data Replication generates SQL CREATE TABLE scripts.

When you enable this option, Data Replication surrounds table names with the following symbols:

- Square brackets [ ] for Microsoft SQL Server
- Apostrophes (') for MySQL
- Double quotation marks (") for other target databases

Note: For Teradata targets, Data Replication surrounds column names with double quotation marks, regardless of this parameter setting.

The default value is false.

schema_converter_change_table_and_column_names_to_default_db_case={true|false}

Indicates whether to change the capitalization of table and column names when Data Replication generates SQL CREATE TABLE scripts.

If you set this parameter to true, Data Replication performs the following case conversion:

- Changes all letters to uppercase for DB2, Netezza, Oracle, and PostgreSQL targets.
- Changes all letters to lowercase for Greenplum targets.
- Does not change the capitalization of table and column names for other target databases.

The default value is true.

Generating Audit Log Tables Based on Target Table Schema

For all target types for which Data Replication supports Merge Apply mode, including Greenplum, Netezza, Oracle, Teradata, and Vertica, you can generate audit log tables based on the schema of existing target

tables. Also, if InitialSync uses the Bulk Copy Program (BCP) to load data to Microsoft SQL Server targets, InitialSync requires audit log tables to process SQL expressions in SQL Apply mode.

Before you begin, open the configuration and connect to the source and target databases.

1. Click **Schema** > **Generate Schema for the Target** on the menu bar, or click the **Generate Schema for the Target** icon button on the Replication Configurations toolbar.

   The **Generating Schema for the Target** dialog box appears.
2. Under **Source/Target**, select the **Target** option.
3. In the **Schema/owner** field, enter the schema or owner name of the target tables that you want to use to generate the audit log tables.

   Tables that match the selected schema or owner name are listed in the **Tables** box.

   Tip: At any time, you can click **Refresh** to refresh the schemas for the source and target databases.
4. To filter the list of tables, in the **Table filter** field, enter the first few letters of the target table names that you want to use for generating the audit log tables. For case-sensitive filtering, enclose the filter value in double quotation marks (").

   Only the tables that match the filter criteria display in the **Tables** box.
5. Select the tables to use for generating the target audit log tables.
   - To select all listed tables, click **Select All**.
   - To select tables individually, select the **Convert** check box in the table row.
6. Under **Target**, in the **Schema** field, enter a schema or owner name.
7. In the **Database type** list, select the target database type.
8. To generate audit log tables, select the **Generate audit log tables** check box. In the adjacent **Audit tables suffix** field, enter a suffix for the generated audit log table names.
   - For SQL Apply and Merge Apply, specify the suffix that is specified for the **Log table suffix for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view.
   - For Audit Apply, you can specify any alphanumeric string as the suffix. However, to use the **Map All** or **Wildcard map** button to quickly map all of the audit log tables to the corresponding source tables, you must enter the suffix that is specified for the **Log table suffix for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view.

   The default suffix is _LOG.
9. To create the audit log tables, perform one of the following actions:
   - To generate a script with SQL CREATE TABLE statements, click **Save**. In the **Save** dialog box, save the generated script to a directory of your choice. You can run the script later, if necessary.
   - To create the audit log tables on the target immediately, click **Run**.

     Note: If you cancel the Run operation while it is in process, Data Replication does not cancel any tables that were already created on the target. You can find the SQL CREATE TABLE statements for these tables in the %DBSYNC_HOME%/logs/ui/se.log file.
10. Click **Close**.

Generating Avro Schemas for Apache Kafka Consumers

For configurations that have Apache Kafka targets, you can generate and save Avro schema files based on source table schema. You can save the generated text files for later use by Kafka consumers that subscribe to the Kafka topics to which Data Replication writes messages.

Note: Data Replication, as a producer application, also requires an Avro schema file. However, this schema is transparently generated for Data Replication use.

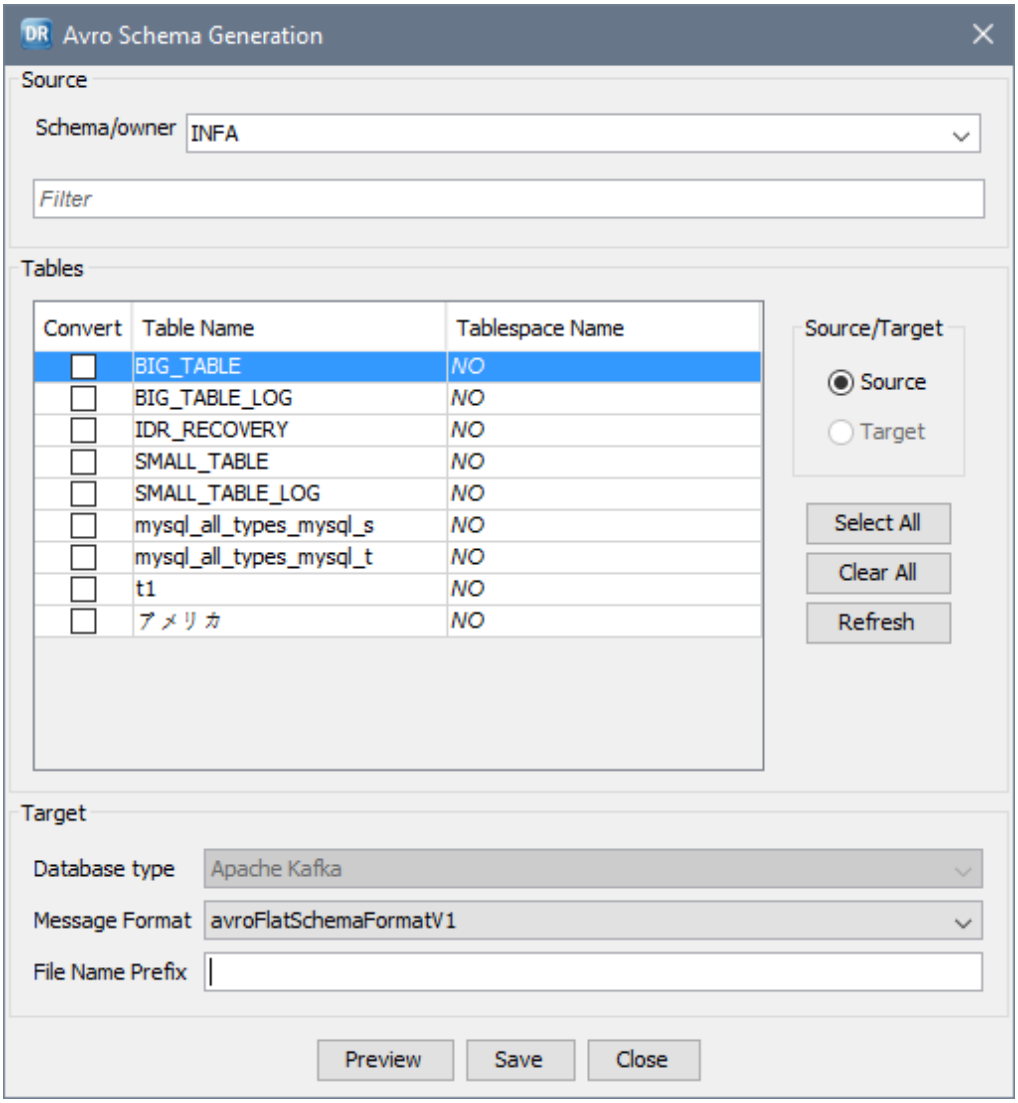
Before you begin, open the configuration and connect to the source database and the Kafka target.

1. Click **Schema** > **Generate Schema for the Target** on the menu bar, or click the **Generate Schema for the Target** icon button on the Replication Configurations toolbar.

   The **Avro Schema Generation** dialog box appears.
2. Under **Source/Target**, verify that the **Source** option is selected.
3. In the **Schema/owner** field, select the schema or owner name for the source tables that you want to use to generate the Avro schema files.

   The tables that match the selected schema or owner name are listed under **Tables**.

The following image shows an example list of tables that match a schema or owner name of INFA:



4. To filter the list of tables, in the *Filter* field, enter the first few letters of the source table names that you want to use for generating Avro schema files. For case-sensitive filtering, enclose the filter value in double quotation marks.

   Only the tables that match the filter criteria are listed under **Tables**.

   Tip: At any time, you can click **Refresh** to refresh the schemas for the source database and Kafka target.

5. Select the tables to use for generating the Avro schema files:
   - To select all listed tables, click **Select All**.
   - To select tables individually, select the **Convert** check box in the table row.

     Tip: When only one table is selected, you can click **Preview** to view a preview of the Avro schema text file that Data Replication will generate for the selected table.

6. Under **Target**, verify that the **Database type** list displays **Apache Kafka**. You cannot edit this field.

7. In the **Message Format** field, select one of the following Avro schema formats to use for formatting messages:
   - **avroFlatSchemaFormatV1**. Use the Avro flat schema format, which lists all Avro fields in one record.
   - **avroNestedSchemaFormatV1**. Use the Avro nested schema format, which organizes each type of information in a separate record.
8. Optionally, in the **File Name Prefix** field, specify a prefix for the Avro schema file names.

   Avro schema file prefixes can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.
9. To generate the Avro schema files, click **Save**. Then in the **Save** dialog box, save the generated schema files to a directory of your choice.

   Data Replication creates a separate text file for each selected source table. The names for the generated text files use the following format: <*preifx*>*_table_name*.txt. You can use this file later with Kafka consumers that subscribe to the Kafka topics to which Data Replication writes messages.

   Note: All source columns are included in the generated schema files. To remove any columns that are not mapped in the configuration, manually edit the generated Avro schema files.
10. At the confirmation prompt, click **OK**.
11. Click **Close**.

Related Topics:

- "Kafka Message Structure" on page 36

Handling Source Tables with Long Table or Column Names

If you have source tables with long table or column names, alternative strategies are available for shortening these names when generating corresponding audit log tables or target tables. Otherwise, the table names on the target might exceed the maximum length that the target database allows.

Merge Apply mode requires a target table and audit log table that correspond to each source table on the target. Audit Apply mode requires an audit log table only. Also, if you use InitialSync with the Bulk Copy Program (BCP) to load data to Microsoft SQL Server targets based on a configuration that includes SQL expressions, InitialSync requires audit log tables to process the SQL expressions in SQL Apply mode.

If you use Merge Apply mode, SQL Apply mode for the special InitialSync with BCP processing case, or Audit Apply mode with table mappings based on wildcard expressions or exact name matching, audit log table names include the suffix that is defined on the **Runtime Settings** tab > **Calculated Columns** view. Also, the before- and after-image columns in the audit log tables have suffixes that are defined on the **Calculated Columns** view. You can shorten the table and column suffixes to comply with name length limits of the target database.

Strategies for Handling Long Table Names

If you have long source table names, use one of the following strategies to create target tables and audit log tables with names that do not exceed the maximum length that the target database allows:

- For the audit log tables, use a log suffix that is shorter than the default suffix of _LOG. For example, use _L or a blank log suffix.

- For Merge Apply and SQL Apply modes, create a script that generates regular target tables and audit log tables with shorter table names. In this case, the Data Replication Console cannot map tables based on table names. You must map these tables manually.

Editing the Audit Log Table Suffix

If a source table name is long and you generate an audit log table name with the default suffix of _LOG based on the source schema, the audit log table name might be too long. You can define a shorter suffix.

Alternatively, you can use a blank log suffix. However, if you use a blank suffix in Merge Apply mode, you must use different schemas to generate regular target tables and audit log tables.

Note: For Netezza targets, you cannot use different schemas to generate regular target tables and audit log tables because a Netezza user can own only one schema. Therefore, do not use a blank suffix for Netezza audit log tables.

1. On the **Runtime Settings** tab > **Calculated Columns** view, replace the default audit table name suffix of _LOG with a shorter suffix in the **Log table suffix for merge apply** field.

   If you use Merge Apply mode, or if you use InitialSync with the Bulk Copy Program (BCP) to load data to SQL Server targets and need to process SQL expressions defined for source tables in SQL Apply mode, Data Replication uses this suffix to determine the audit log table that corresponds to each mapped target table.

   In Audit Apply mode, the Data Replication Console uses this suffix only if you map the source tables to audit log tables based on wildcard expressions or exact name matching.
2. Generate audit log tables based on the source table schema.
3. If you use Merge Apply mode and a blank log suffix, specify a schema that includes audit log tables in the **Log table schema for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view.
4. Save the configuration.

Related Topics:

- "Generating Target Tables and Audit Log Tables Based on Source Table Schema" on page 199

Manually Editing Target Table Names in an SQL Script

If you have source tables with long table names that exceed the length limit that the target database allows, you can edit the table names in an SQL script that you use to generate the regular target tables. After you run this script, you can generate audit log tables based on the target table schema if you require these tables for Merge Apply processing or for InitialSync with the Bulk Copy Program (BCP) processing of SQL expressions in SQL Apply mode for Microsoft SQL Server targets.

Because the source and target table names do not match, the Data Replication Console cannot map source and target tables based on table names. You must manually map the source tables to the generated target tables.
For example, in the SQL script, replace the OR_MEDICAL_DEVICES_FOR_OPERATION target table name with OR_MED_DEVS_FOR_OPERATION. Then generate the audit log table using the default suffix of _LOG. The following table names are used for the replication:

- The source table name is OR_MEDICAL_DEVICES_FOR_OPERATION.
- The target table name to which the source table is mapped is OR_MED_DEVS_FOR_OPERATION.
- The audit log table name that corresponds to the target table is OR_MED_DEVS_FOR_OPERATION_LOG.

1. Create a script for generating target tables and save it on the local file system.

2. Open the generated script in a text editor and replace the long table names with shorter names.

   Ensure that you can append the audit log table suffix that is specified in the **Log table suffix for merge apply** field on the **Runtime Settings** > **Calculated Columns** view to new names without exceeding the name length limit of the target database.
3. Save the edited script.
4. Run the script to generate the target tables.
5. Generate the audit log tables based on the target table schema.
6. On the **Map Tables** tab, manually map the source tables to the corresponding target tables.
7. Save the configuration.

Related Topics:

- "Generating Audit Log Tables Based on Target Table Schema" on page 201

Strategies for Handling Long Column Names

If you have source tables with long column names, alternative strategies are available for handling the long column names so that the corresponding column names on the target do not exceed the maximum length that the target database allows.

Use one of the following strategies to shorten column names in generated audit log tables or target tables on the target:

- For audit log table columns that contain before- and after-images, use suffixes that are shorter than the default suffixes of _OLD and _NEW. For example, you can use _O and _N as the suffixes.
- For Merge Apply or SQL Apply mode, create a script that generates regular target tables or audit log tables with shorter column names. In this case, the Data Replication Console cannot map columns based on column names. You must map these columns manually.
- For replications that use Merge Apply mode with Teradata targets, create views that have shorter column names for the target tables and generate audit log tables based on these views. The column names in the views are truncated, but the Data Replication Console ensures that the names are unique. When defining mappings, map the views to the source tables.

Editing the Suffixes for Before-Image and After-Image Columns in Audit Log Tables

If a generated audit log table has before-image and after-image column names that are too long with the default suffixes of _OLD and _NEW, define shorter suffixes.

1. On the **Runtime Settings** > **Calculated Columns** view, replace the default before-image and after-image suffixes in the **Before image column name ending** and **After image column name ending** fields with shorter suffixes.

   For example, enter _O and _N.
2. Generate the audit log tables based on the source table schema.
3. Save the configuration.

Related Topics:

- "Generating Target Tables and Audit Log Tables Based on Source Table Schema" on page 199

Manually Editing Target Column Names in an SQL Script

If you have source columns with long column names that exceed the length limit that the target database allows, you can edit the column names in an SQL script that you use to generate the target tables. After you run this script, you can generate the audit log tables based on the target table schema if you require these tables for Merge Apply processing or for InitialSync with the Bulk Copy Program (BCP) processing of SQL expressions in SQL Apply mode for Microsoft SQL Server targets.

Because the source and target column names do not match, the Data Replication Console cannot map columns based on column names. You must manually define column mappings for the generated tables.

For example, in the SQL script, replace the TOTAL_NUMBER_OF_MEDICAL_CLAMPS target table name with TTL_NUM_OF_MED_CLAMPS. Then generate an audit log table based on the target table schema and use the default suffixes _OLD and _NEW for the before-image and after-image columns. The Data Replication Console creates the TTL_NUM_OF_MED_CLAMPS_OLD and TTL_NUM_OF_MED_CLAMPS_NEW columns. Map the TOTAL_NUMBER_OF_MEDICAL_CLAMPS source column to the TTL_NUM_OF_MED_CLAMPS target column.

1. Create a script for generating regular target tables and save it on the local file system.
2. Open the generated script in a text editor and replace the long column names with shorter names.

   Ensure that you can append the before image and after image suffixes that are specified on the **Runtime Settings** > **Calculated Columns** view to new names without exceeding the name length limit of the target database.
3. Save the edited script.
4. Run the script to generate target tables.
5. Generate the audit log tables based on the target table schema.
6. On the **Map Tables** tab, manually map the source columns to the corresponding target columns.
7. Save the configuration.

Related Topics:

- "Generating Audit Log Tables Based on Target Table Schema" on page 201

Using Views to Generate Audit Log Tables and Replicate Data to Teradata Targets

If you use Merge Apply mode with Teradata targets and have source tables with long column names, the Data Replication Console enables you to create views with shorter column names. You can then use the views to generate the audit log tables and to create mappings.

First generate a target table that has the same column names as the source tables. For the generated target table, create a view that includes the columns that correspond to the source columns but that have truncated column names. Generate the audit log table based on the view. Then map the source table to the corresponding view. You must perform all of these tasks in a single Data Replication Console session.

For example, a source table contains the TOTAL_NUMBER_OF_MEDICAL_CLAMPS column. You generate a target table with a column of the same name. You create a view for the target table that includes the column with the truncated name of TOTAL_NUMBER_OF_MEDICAL_C1. You generate an audit log table based on the

view. The audit log table includes the TOTAL_NUMBER_OF_MEDICAL_C1_OLD and TOTAL_NUMBER_OF_MEDICAL_C1_NEW columns.

1. Generate the target table for the source table that has the long column names.
2. On the **Map Tables** view, right-click the target table and click **Create View**.
3. Enter a name for the view and click **OK**.

   The view appears in the list of target tables.
4. Generate an audit log table based on the view.
5. Map the source table to the view.
6. Save the configuration.

Mapping Source and Target Tables

You must map source tables to target tables for a replication configuration.

First connect to the source database from the **Source Database** tab and connect to the target database from the **Target Database** tab.

You can map source tables in a single schema to target tables in a single schema. Then repeat the mapping process to map tables for other pairs of schemas.

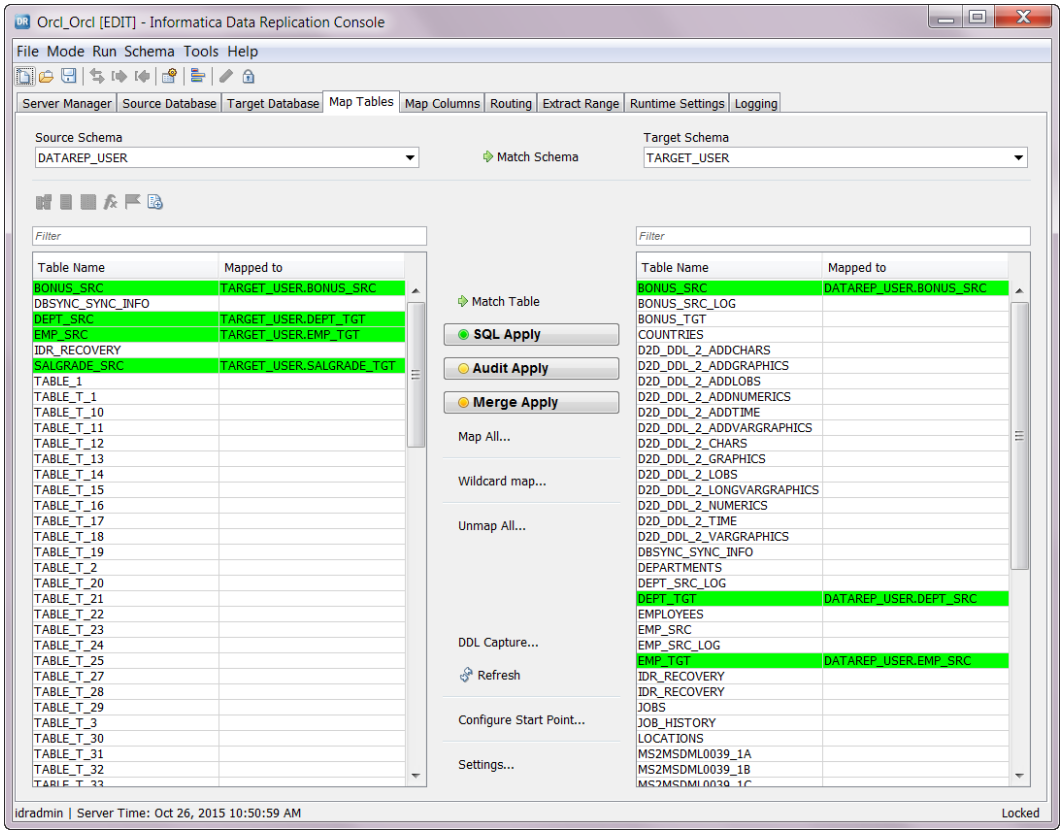
Depending on the target type, use one of the following apply modes to map tables:

- **SQL Apply**. You can use SQL Apply if you have DB2, Microsoft SQL Server, MySQL, PostgreSQL, or Oracle targets. These mappings are highlighted in green.
- **Audit Apply**. You must use Audit Apply if you have Apache Kafka, Cloudera, Flat File, or Hortonworks targets. You can optionally use Audit Apply for relational target types. These mappings are highlighted in yellow.
- **Merge Apply**. You must use Merge Apply if you have Greenplum, Netezza, Teradata, or Vertica targets. You can optionally use Merge Apply for Oracle targets. These mappings are highlighted in orange.

To create a mapping, use any of the following methods:

- Map tables individually.
- Map multiple tables based on exact table-name matching.
- Map tables based on wildcard expressions for table names.

The following image shows SQL Apply mappings on the **Map Tables** tab:



The list boxes on the **Map Tables** tab list the objects for which Data Replication supports replication. The following limitations apply:

- For Microsoft SQL sources and targets, the **Map Tables** tab does not display views.
  To display indexed views, you can set the show_views parameter to true in the *DataReplication_installation*/uiconf/default.cfg file.

  Note: If the default.cfg file does not exist, use a text editor to create the file in the uiconf directory and then add the show_views parameter in the file. For more information, see "Default.cfg File" on page 391.
- For Oracle sources and targets, the **Map Tables** tab does not display temporary tables.

Mapping Source and Target Tables Individually

You can map source and target tables that have compatible schema individually.

1. Click the **Map Tables** tab.
2. In the **Source Schema** list, select a source schema.

   Note: For Microsoft SQL Server sources, this list displays SQL Server databases, which are similar to schemas of other source types.

   A list of source tables in the specified schema appears in the left list box. To filter the list, enter the first few letters of a table name in the **Filter** field. For case-sensitive filtering, enclose the filter value in double quotation marks.
3. Select the target schema in one of the following ways:
   - In the **Target Schema** list, select the target schema.

Note: For Microsoft SQL Server targets, this list displays SQL Server databases.

- Click the **Match Schema** button to use the source schema for the target.

A list of target tables that use the specified schema appears in the right list box. To filter the list, enter the first few letters of a table name in the **Filter** field. For case-sensitive filtering, enclose the filter value in double quotation marks.
For Apache Kafka, Cloudera, Flat File, or Hortonworks targets, you do not need to perform this step. The Data Replication Console automatically displays the source schema in the **Target Schema** field and lists the source table names on the target side.

Tip: At any time, you can click **Refresh** to refresh the schemas for the source and target databases. If you cancel the refresh operation while it is in progress, Data Replication marks the configuration as not valid. Also, you cannot save the configuration unless you click **Refresh** again and do not cancel the refresh operation.

4. Select a source table.

   The selected table is highlighted in red.

5. Select the target table in one of the following ways:

   - Click the target table.
   - If the target table name matches the selected source table name exactly, click **Match Table**.

   If you plan to use Audit Apply mode, you must select the corresponding audit log table in the target database.

   For Apache Kafka, Cloudera, Flat File, or Hortonworks targets, which do not use tables, you do not need to perform this step. When you select a source table, the source table name is automatically displayed and selected on the target side as a logical mapping.

   Tip: To undo a table selection, right-click the selected table row and select **Clear Selection**.

6. To map the selected source table to the selected target table, click the one of the following buttons, as appropriate for the target type:

   - **SQL Apply**
   - **Audit Apply**
   - **Merge Apply**

   Tip: To remove one or more table mappings, select each source table row and click the **Remove Mapping** icon button. To remove a single mapping, you can right-click the source table row and click **Remove Mapping**, or double-click the source table row. To undo all mappings, click **Unmap All**. After you remove mappings by using any of these methods, click **Yes** in the confirmation box.

7. Repeat steps 4 to 6 for each pair of source and target tables that you want to map.
8. To map tables in different schemas, select another source schema and another target schema. Then repeat steps 4 through 7.

From the **Map Tables** tab, you can also enable the replication of Create Table and Drop Table DDL operations at the schema level, customize the types of DDL and DML changes to apply by target table, configure start points for the Extractor and Applier tasks, add Tcl and SQL expressions, and configure conflict resolution.

Related Topics:

- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Configuring the Start Points for Extractor and Applier Tasks" on page 226
- "Customizing Apply Settings for Target Tables" on page 225
- "Adding Tcl and SQL Expressions" on page 243
- "Configuring Conflict Resolution" on page 233

Mapping Source and Target Tables Based on Exact Name Matching

You can map source and target tables based on exact table name matching.

For the Merge Apply and SQL Apply modes, this method maps the source tables to target tables that have identical names. For the Audit Apply mode, this method maps the source tables to the audit log tables that have matching table names followed by the audit log table suffix.

You can map all of the source tables for which target tables with the same names exist, or use the Shift or Ctrl keys to select a subset of source tables to map to target tables with the same names.

1. Click the **Map Tables** tab.
2. In the **Source Schema** list, select a source schema.

   Note: For Microsoft SQL Server targets, this list displays SQL Server databases, which are similar to schemas of other source types.

   A list of source tables that use the specified schema appears in the left list box. To filter the list, enter the first few letters of a table name in the **Filter** field. For case-sensitive filtering, enclose the filter value in double quotation marks.
3. Select the target schema in one of the following ways:
   - In the **Target Schema** list, select the target schema.

     Note: For Microsoft SQL Server targets, this list displays SQL Server databases.
   - Click **Match Schema** to use the source schema for the target.

   A list of target tables that use the schema appears in the right list box. To filter the list, enter the first few letters of a table name in the **Filter** field. For case-sensitive filtering, enclose the filter value in double quotation marks.
   For Apache Kafka, Cloudera, Flat File, or Hortonworks targets, you do not need to perform this step. The Data Replication Console automatically displays the source schema in the **Target Schema** field and lists the source table names on the target side.

   Tip: At any time, you can click **Refresh** to refresh the schemas for the source and target databases. If you cancel the refresh operation while it is in progress, Data Replication marks the configuration as not valid. Also, you cannot save the configuration unless you click **Refresh** again and do not cancel the refresh operation.
4. To map all source tables for which target tables with the same names exist, perform the following steps:
   a. Click **Map All**.
   b. In the **Map All** dialog box, select one of the following apply modes to use for the table mappings:
      - **SQL Apply**
      - **Audit Apply**
      - **Merge Apply**

      If you select **Audit Apply**, the Data Replication Console looks for audit log tables that have matching table names followed by the suffix that is specified on the **Runtime settings** tab > **Calculated Columns** view. The default suffix is _LOG.

      Tip: To exit the **Map All** dialog box without mapping the tables, click **Cancel**.
   c. Click **Map**.

      Tip: To clear all mappings, click **Unmap All**.

5. To map a subset of the source tables to target tables with the same names, use the Shift or Ctrl key to select the source tables and click **Match Table**. Then click the **SQL Apply**, **Audit Apply**, or **Merge Apply** button.

6. Repeat steps 1 through 5 for each pair of source and target schema for which you want to define table mappings based on exact name matching.

From the **Map Tables** tab, you can also enable the replication of Create Table and Drop Table DDL operations at the schema level, customize the types of DDL and DML changes to apply by target table, configure start points for the Extractor and Applier tasks, add Tcl and SQL expressions, and configure conflict resolution.

Related Topics:

- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Configuring the Start Points for Extractor and Applier Tasks" on page 226
- "Customizing Apply Settings for Target Tables" on page 225
- "Adding Tcl and SQL Expressions" on page 243
- "Configuring Conflict Resolution" on page 233

Mapping Source and Target Tables Based on Wildcard Expressions

You can map source and target tables based on wildcard expressions. If the mapped tables have different column names, you can also map the columns based on wildcard expressions.

A wildcard expression includes one or more asterisk (*) wildcard characters that represent zero or more characters. The wildcard characters can occur anywhere in a table or column name.

Use wildcard expressions to define sets of source and target tables or columns to map. The Data Replication Console maps the selected source tables or columns to the selected target tables or columns by matching the substrings in their names that are represented by the asterisk (*) in the wildcard expression.

1. Click the **Map Tables** tab.
2. In the **Source Schema** list, select a source schema.

   Note: For Microsoft SQL Server sources, this list displays SQL Server databases, which are similar to schemas of other source types.

   A list of source tables that use the specified schema appears in the left list box. To filter the list, enter the first few letters of a table name in the *Filter* field. For case-sensitive filtering, enclose the filter value in double quotation marks.
3. Select the target schema in one of the following ways:
   - In the **Target Schema** list, select the target schema.

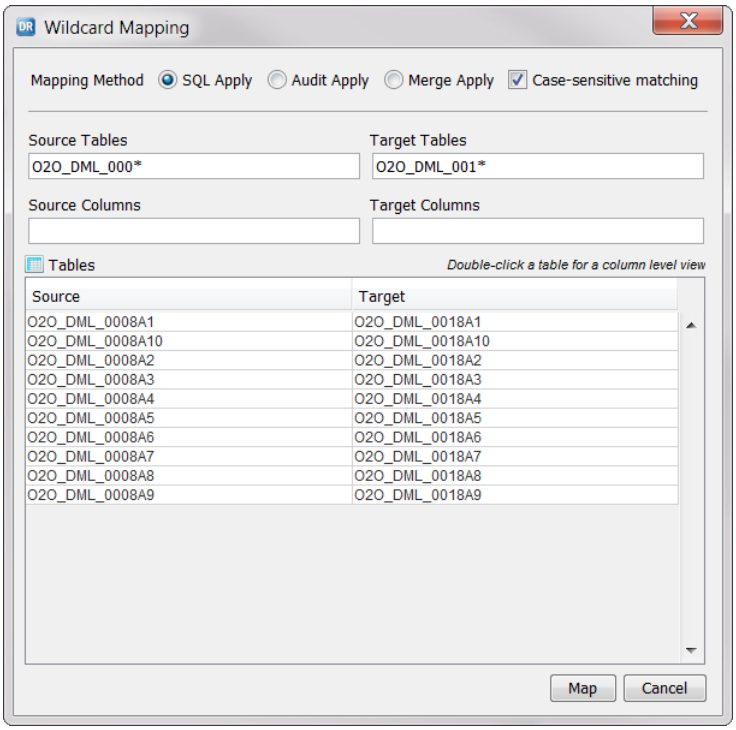
     Note: For Microsoft SQL Server targets, this list displays SQL Server databases.
   - Click **Match Schema** to use the source schema for the target.

   A list of target tables that use the specified schema appears in the list box on the right. To filter the list, enter the first few letters of a table name in the *Filter* field. For case-sensitive filtering, enclose the filter value in double quotation marks.
   For Apache Kafka, Cloudera, Flat File, or Hortonworks targets, you do not need to perform this step. The Data Replication Console automatically displays the source schema in the **Target Schema** field and lists the source table names on the target side.

Tip: At any time, you can click **Refresh** to refresh the schemas for the source and target databases. If you cancel the refresh operation while it is in progress, Data Replication marks the configuration as not valid. You will not be able to save the configuration unless you click **Refresh** again and wait for the refresh operation to complete.

4. To map tables based on wildcard expressions, click **Wildcard map**.

   The **Wildcard Mapping** dialog box appears:



5. For the **Mapping Method**, select one of the following options that reflect the apply mode:
   - **SQL Apply**
   - **Audit Apply**
   - **Merge Apply**
6. If you do not want to use case-sensitive matching against wildcard expressions, clear the **Case-sensitive matching** check box.
7. In the **Source Tables** and **Target Tables** fields, enter wildcard expressions. In the expressions, you can use one or more asterisk (*) wildcard characters to represent the portions of the source and target table names that must match to map the tables.

   Use the following guidelines:

   - For Microsoft SQL Server databases, include the schema name in the wildcard expression to avoid incorrect mappings of tables that are in different databases but have identical table names.
   - If the source and target table names are identical, enter only the asterisk (*) wildcard character in the **Source Tables** and **Target Tables** fields. You can then enter more specific wildcard expressions for the column names in the **Source Columns** and **Target Columns** fields.
   - If you selected **Audit Apply**, enter the wildcard expression in the **Target Tables** field without the audit log table suffix that is specified on the **Runtime Settings** tab > **Calculated Columns** view.

   Note: Each asterisk (*) in a wildcard expression for source tables defines a capturing group. A *capturing group* stores the part of a table name that is represented by the asterisk. You can use back references to

capturing groups in the wildcard expressions for target tables. Use the $*n* wildcard character in a wildcard expression for target tables to insert a back reference to a source capturing group, where *n* is the sequence number of the capturing group. Capturing groups in a wildcard expression are numbered from left to right.

The **Wildcard Mapping** dialog box displays the matching source and target tables in the **Tables** box.

8. If the matching tables have different column names, enter wildcard expressions for the source and target columns in the **Source Columns** and **Target Columns** fields.

   If you selected **Audit Apply**, enter a wildcard expression in the **Target Columns** field without the before-image and after-image suffixes that are specified on the **Runtime Settings** tab > **Calculated Columns** view.

9. If you specified wildcard expressions for the column names, review the column matches for each pair of source and target tables that matched.

   To view the column matches, double-click the row for the matching tables. To switch back to the table matches, click the **Back** button.

10. In the **Tables** box, select one or more rows for the table matches that you want to map.

    If you select none of the rows, the Data Replication Console maps all of the listed tables.

    Tip: Double-click a row to view the columns in the matched source and target tables.

11. Click **Map** to map the matching tables and columns.

12. Repeat steps 2 through 11 for each pair of source and target schema for which you want to define table and column mappings with wildcards.

From the **Map Tables** tab, you can also enable the replication of Create Table and Drop Table DDL operations at the schema level, customize the types of DDL and DML changes to apply by target table, configure start points for the Extractor and Applier tasks, add Tcl and SQL expressions, and configure conflict resolution.

Related Topics:

- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Configuring the Start Points for Extractor and Applier Tasks" on page 226
- "Customizing Apply Settings for Target Tables" on page 225
- "Adding Tcl and SQL Expressions" on page 243
- "Configuring Conflict Resolution" on page 233

Examples of Wildcard Expressions for Table and Column Mappings

Review the following examples to learn how to define table and column mappings for different apply modes based on wildcard expressions. You define the mappings in the **Wildcard Mapping** dialog box.

Example 1. Wildcard Expressions for an SQL Apply Mapping of Tables

Map the following source tables and columns to Microsoft SQL Server target tables and columns in SQL Apply mode:

- TABLE1 on the source and dbo.TABLE1_TGT on the target.
  - COL1 on the source and COL1 on the target.
  - COL2 on the source and COL2 on the target.
- TABLE2 on the source and dbo.TABLE2_TGT on the target.
  - COL1 on the source and COL1 on the target.

- COL2 on the source and COL2 on the target.

The SQL Server target table names include the owner name dbo and the suffix _TGT.

Complete the following steps:

1. In the **Mapping Method** field, select **SQL Apply**.
2. In the **Source Table** field, enter the `*` wildcard only.
3. In the **Target Tables** field, enter dbo.`*_TGT` to select all target tables of the dbo owner with table names that end with the _TGT.

   In the list box at the bottom of the **Wildcard Mapping** dialog box, the Data Replication Console matches TABLE1 to dbo.TABLE1_TGT and matches TABLE2 to dbo.TABLE2_TGT.
4. To map these tables, click **Map**.

Note: You do not need to enter wildcard expressions for the column mappings because the source and target tables have identical column names.

Example 2. Wildcard Expressions for a Merge Apply Mapping of Tables and Columns

Map the following source and target tables and columns in Merge Apply mode:

- TABLE_SRC_1 on the source and TABLE_TGT_1 on the target.
  - COL_SRC_1 on the source and COL_TGT_1 on the target.
  - COL_SRC_2 on the source and COL_TGT_2 on the target.
- TABLE_SRC_2 on the source and TABLE_TGT_2 on the target.
  - COL_SRC_1 on the source and COL_TGT_1 on the target.
  - COL_SRC_2 on the source and COL_TGT_2 on the target.

Use wildcard expressions for both the table and column mappings.

Complete the following steps:

1. In the **Mapping Method** field, select **Merge Apply**.
2. In the **Source Tables** field, enter `*SRC*` to select all source tables with table names that include SRC in the middle of the names.
3. In the **Target Tables** field, enter `*TGT*` to select all target tables with table names that include TGT in the middle of the names.

   The Data Replication Console matches TABLE_SRC_1 to TABLE_TGT_1 and matches TABLE_SRC_2 to TABLE_TGT_2.
4. In the **Source Columns** field, enter `*SRC*`.
5. In the **Target Columns** field, enter `*TGT*`.

   In the list box at the bottom, double-click the TABLE_SRC_1 row to view column matches. Then view columns matches for TABLE_SCR_2 in the same way.
6. Click **Map**.

Example 3. Wildcard Expressions for an Audit Apply Mapping of Tables

Map the following source tables to the target audit log tables in Audit Apply mode. The source and target table names have different prefixes, and the target audit log table names include the _LOG suffix that is specified on the **Runtime Settings** tab > **Calculated Columns** view.

- SRC_TABLE1 on the source and TGT_TABLE1_LOG on the target.
  - COL1 on the source and a pair of COL1_NEW and COL1_OLD columns on the target.

- COL2 on the source and a pair of COL2_NEW and COL2_OLD columns on the target.

- SRC_TABLE2 on the source and TGT_TABLE2_LOG on the target.
  - COL1 on the source and a pair of COL1_NEW and COL1_OLD columns on the target.
  - COL2 on the source and a pair of COL2_NEW and COL2_OLD columns on the target.

Complete the following steps:

1. In the **Mapping Method** field, select **Audit Apply**.
2. In the **Source Table** field, enter `SRC*` to select all source tables with table names that begin with SRC.
3. In the **Target Tables** field, enter `TGT*` to select the audit log tables with table names that begin with TGT and end with the suffix _LOG.

   The Data Replication Console matches SRC_TABLE1 to TGT_TABLE1_LOG and matches SRC_TABLE2 to TGT_TABLE2_LOG.
4. To map these tables, click **Map**.

Example 4. Wildcard Expressions for an SQL Apply Mapping of Tables

Map the following source and target tables and columns in SQL Apply mode. The tables have identical table names, but the columns have different column names.

- TABLE1 on the source and TABLE1 on the target.
  - COL1_SRC on the source and COL1_TGT on the target.
  - COL2_SRC on the source and COL2_TGT on the target.
- TABLE2 on the source and TABLE2 on the target.
  - COL1_SRC on the source and COL1_TGT on the target.
  - COL2_SRC on the source and COL2_TGT on the target.

Use wildcard expressions for both the table and column mappings.

Complete the following steps:

1. In the **Mapping Method** field, select **SQL Apply**.
2. In the **Source Table** and **Target Tables** fields, enter `*` to map source and target tables that have identical names.

   The Data Replication Console matches the TABLE1 source table to the TABLE1 target table and matches the TABLE2 source table to the TABLE2 target table.
3. In the **Source Columns** field, enter `*SRC`.
4. In the **Target Columns** field, enter `*TGT`.

   In the list box at the bottom, double-click the TABLE1 row to view column matches. Then view column matches for TABLE2 in the same way.
5. To map these tables, click **Map**.

Example 5. Wildcard Expressions with Capturing Groups for an SQL Apply Mapping of Table and Columns

Map the following source and target tables and columns in SQL Apply mode:

- SRC_TABLE1_GRP1 on the source and GROUP1_TABLE1_TGT on the target.
  - COL1 on the source and COL1 on the target.
  - COL2 on the source and COL2 on the target.

- SRC_TABLE2_GRP3 on the source and GROUP3_TABLE2_TGT on the target.
  - COL1 on the source and COL1 on the target.
  - COL2 on the source and COL2 on the target.

Complete the following steps:

1. In the **Mapping Method** field, select **SQL Apply**.
2. In the **Source Tables** field, enter `SRC_*_GRP*`.
3. In the **Target Tables** field, enter `GROUP$2_$1_TGT`.

   The $2 wildcard matches against the capturing group that is represented by the second asterisk in the **Source Tables** field, which is a group ID. The $1 wildcard matches against the capturing group that is represented by the first asterisk, which is a table ID. As a result, the target table matching strings contain the group and table IDs in the reverse order from the source matching strings.

   Based on these source and target expressions, the Data Replication Console matches SRC_TABLE1_GRP1 to GROUP1_TABLE1_TGT and matches SRC_TABLE2_GRP3 to GROUP3_TABLE2_TGT.
4. Click **Map**.

Note: You do not need to enter wildcard expressions for the column mappings because the source and target tables have identical column names.

Mapping Validation

When you create a replication configuration, the Data Replication Console validates table, view, and column mappings.

By default, the Data Replication Console does not allow you to map a source table to a target in the following circumstances:

- The source table type is not supported.
- The source table contains one or more columns that have unsupported datatypes.

Also, if you add virtual columns with an SQL or Tcl expression to a source, the Data Replication Console does not allow you to map the virtual columns to target columns that have incompatible datatypes.

You can use the compatibility_checking_level parameter in the *DataReplication_installation*/uiconf/default.cfg file to change the default mapping validation behavior. Options are:

- **0**. Disable mapping validation.
- **1**. Validate only column mappings, including virtual column mappings.
- **2**. Validate only table mappings.
- **3**. Validate table and column mappings. Map tables and views even if they include columns with unsupported datatypes. Map only the columns that have supported datatypes.
- **4**. Validate table and column mappings. Do not map tables and views that include columns with unsupported datatypes. Also, do not map virtual source columns to target columns that have incompatible datatypes.

Default value: 4

Note: If the default.cfg file does not exist, use a text editor to create this file in the uiconf directory and then add the compatibility_checking_level parameter in the file. For more information, see "Default.cfg File" on page 391.

Database Objects That Cannot Be Mapped

The Data Replication Console does not allow the mapping of source tables or views that have certain characteristics unless you use the compatibility_checking_level parameter to override the default mapping validation behavior.

For DB2 for Linux, UNIX, and Windows sources, the following types of tables, views, and columns are not supported in table mappings:

Tables and Views	Columns
- Temporary tables - Views	- Columns that have unsupported datatypes

For Microsoft SQL Server sources, the following types of tables, views, and columns are not supported in table mappings:

Databases	Tables and Views	Columns
- LOB/Backup compression - Database encryption	- Temporary tables - System tables - Wide tables - Materialized and non-materialized views	- Virtual computer columns - Columns that have unsupported datatypes

For MySQL sources, the following types of tables, views, and columns are not supported in table mappings:

Tables and Views	Columns
- Temporary tables - Tables that include spatial datatypes such as GEOMETRY - Views	- Columns that have unsupported datatypes - Identity columns

For Oracle sources, the following types of tables, views, and columns are not supported in table mappings:

Tables and Views	Columns
- Clustered tables - Partitioned index-organized tables defined with both the OVERFLOW and INCLUDING clauses - External tables - Temporary tables - Tables defined with the ROWDEPENDENCIES clause - Non-materialized views	- Hybrid columnar compression (HCC) - User-defined types (UDTs) - Columns that have unsupported datatypes - Columns defined as invisible - Identity columns - Columns that have extended datatypes

For the list of unsupported datatypes, see the *Informatica Datatype Mapping Reference*.

Related Topics:

- "Replication Considerations for DB2 for Linux, UNIX, and Windows Sources" on page 63
- "Replication Considerations for Microsoft SQL Server Sources" on page 68
- "Replication Considerations for Oracle Sources" on page 80

Defining Source Table Indexes

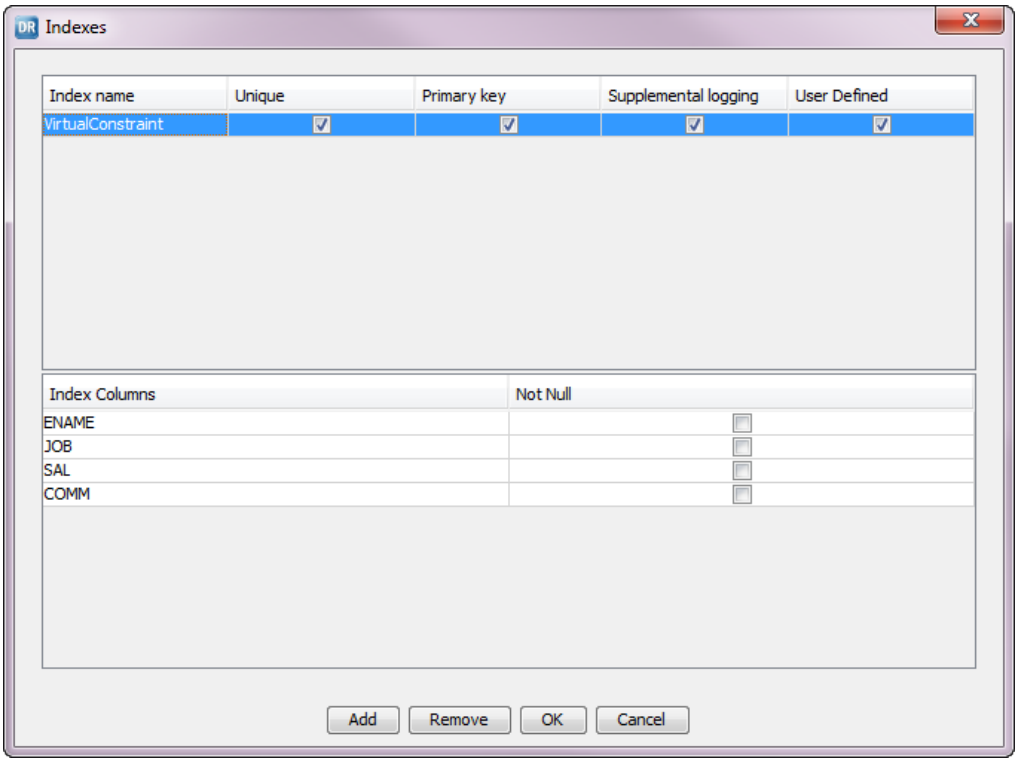
Each source table must have a primary key or unique index definition to perform accurate replication. If a source table has no primary key or unique index definition, or if you do not map these columns in the replication configuration, you can define virtual indexes to uniquely identify rows in target tables.

Editing an Index for a Source Table

You can optionally edit indexes that are defined for a mapped source table.

1. On the **Map Tables** tab, select a mapped source table. Then click the **Show Indexes** icon button above the source tables filter field, or right-click a mapped source table and click **Show Indexes**.

   The **Indexes** dialog box appears.



   The upper list box lists all of the indexes that are defined for the source table. If you select an index row in the upper list box, the index column or columns appear in the lower list box. By default, index columns are displayed in the order in which they are defined in the source table.

   Tip: You can click the **Index Columns** heading to sort the list of index columns in alphabetical order. Click the column heading a second time to sort the list in reverse alphabetical order.

2. In the upper list box, edit the parameters in an index row, as needed.

The following table describes the index parameters:

Parameter	Description
Unique	Select this option if the index is unique.
Primary key	Select this option to use the index as a primary key. You can specify only one primary key for a table. If you select more than one primary key, the Informatica Data Replication Console displays an error and ignores the current primary key.
Supplemental logging	Select this option to enable supplemental logging for the index columns. Data Replication enables supplemental logging for all index columns by default. If you use SQL Apply mode, supplemental logging for columns that are mapped to target index columns help preserve the proper order of updates for multi-threaded apply processing.
User defined	Select this option to enable user-defined mode. In user-defined mode, you can set the Unique, Primary key, and Supplemental logging parameters to values that are different from those that are defined for the index in the source database. You might need to change these parameters to correctly replicate changes to the target. User-defined parameters have priority over the index definitions in the source database and do not affect those definitions. If you revert a user-defined index, all of the index parameters revert to the index definitions in the source database.

Note: You cannot edit the **Not Null** check box in the lower list box. This check box indicates whether or not the column can contain nulls.

3. Click **OK**.

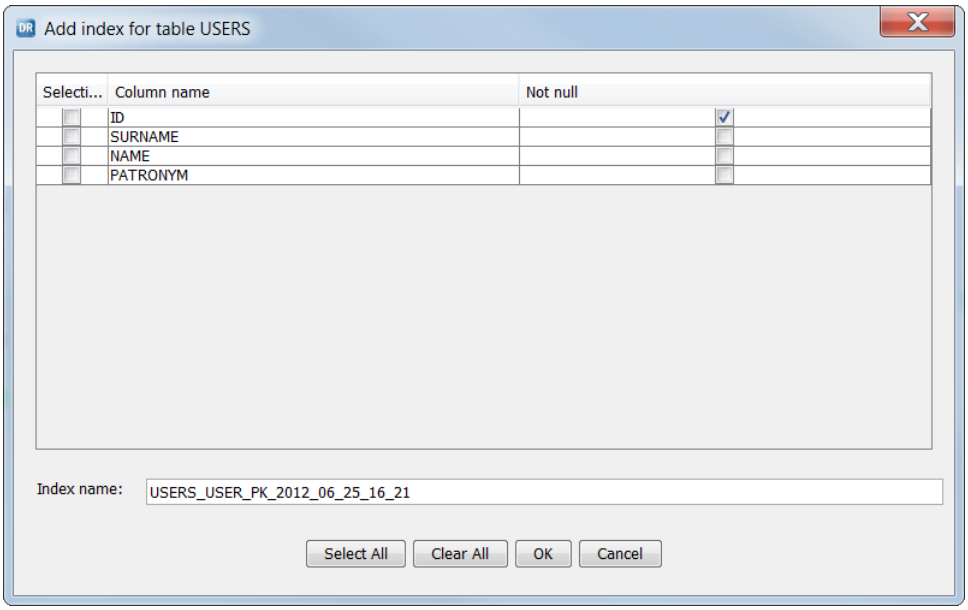
Adding a Virtual Index

If the source database table has no database-level primary key definition, you can define virtual indexes for the source database table to perform accurate replication.

1. On the **Map Tables** tab, select a mapped source table for which to add a virtual index.
2. Click the **Show Indexes** icon button above the source tables filter field, or right-click a mapped source table and click **Show Indexes**.

   The **Indexes** dialog box appears and lists the indexes that are defined for the table.
3. Click **Add**.

The **Add index for table <table name>** dialog box appears.



4. Select the columns to use for the virtual index, or click **Select All** to select all of the columns.

   Note: You cannot edit the **Not null** setting.

5. In the **Index name** field, enter a name for the index or accept the default name.

   Index names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.

6. Click **OK**.

   The **Indexes** dialog box lists the new index.

Removing a Virtual Index

In the Data Replication Console, you can remove a virtual index that you previously defined for a source database table.

1. On the **Map Tables** tab, select the mapped source table from which to remove the virtual index.
2. Click the **Show Indexes** icon button above the source tables filter field, or right-click a mapped source table and click **Show Indexes**.

   The **Indexes** dialog box appears and lists all of the indexes that are defined for the table.

3. Select the index to remove in the upper list box and click **Remove**.

Enabling Replication of DDL Changes at the Schema and Table Levels

You can replicate many types of DDL changes that occur on the source. From the **Map Tables** tab, you can enable replication of Create Table and Drop Table operations for a schema mapping. From the **Map Columns** table, you can enable replication of several other types of DDL operations for a specific table mapping.

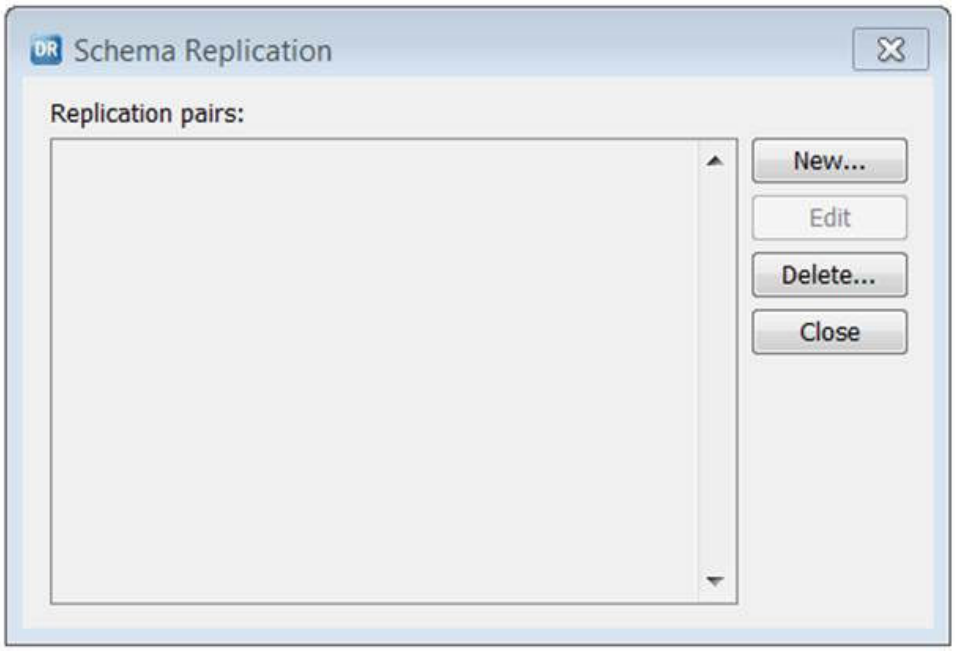
For information about the DDL operations that are supported for each source type, see the following topics:

- "Supported DDL Operations for DB2 for Linux, UNIX, and Windows Sources" on page 65
- "Supported DDL Operations for Microsoft SQL Server Sources" on page 73
- "Supported DDL Operations for MySQL Sources" on page 77
- "Supported DDL Operations for Oracle Sources" on page 85

Note: Data Replication does not support replication of DDL changes for configurations with Apache Kafka targets, regardless of the source type.

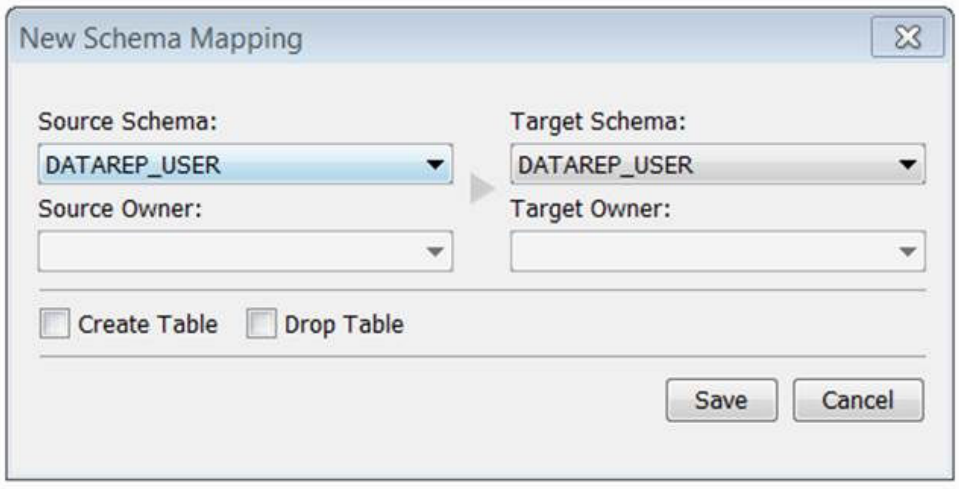
1. To enable replication of DDL operations at the schema-level, perform the following steps:
   a. On the **Map Tables** tab, click **DDL Capture**.

      The **Schema Replication** dialog box appears.



   b. Click **New** to define a schema mapping for DDL replication.

The **New Schema Mapping** dialog box appears.



c. Select a source schema and target schema. You can also select a source owner and target owner.

d. To indicate the type of DDL operation to replicate for the schema mapping, select one or both of the following check boxes:

- **Create Table**
- **Drop Table**

e. Repeat steps b through d for each schema mapping.

Restriction: Do not define multiple schema mappings that use the same source schema and different target schemas. Also, do not use the same schema as the source schema in one mapping and as the target schema in another mapping.

f. Click **Save**.

For configurations with multiple targets, if you override the schema names for secondary targets on the **Routing** tab, Data Replication uses these override schema names for DDL replication.

2. On the **Map Columns** tab, select one or more of the following check boxes under **DDL Capture** to indicate the types of DDL operations to replicate for a specific table mapping:

- **Add Column**
- **Drop Column**
- **Alter Column**
- **Truncate Table**
- **Add Index**
- **Drop Index**

Note: You can also control which of these DDL operations are applied for each target table in the configuration. Click **Settings** on the **Map Tables** tab.

Related Topics:

- "Customizing Apply Settings for Target Tables" on page 225
- "DDL Replication" on page 53
- "DDL Replication Considerations" on page 54
- "Customizing Column Mappings" on page 240
- "CREATE TABLE and ADD COLUMN Operations" on page 55

- "Datatype Conversion Rules" on page 52
- "Replication of TRUNCATE TABLE Operations" on page 57

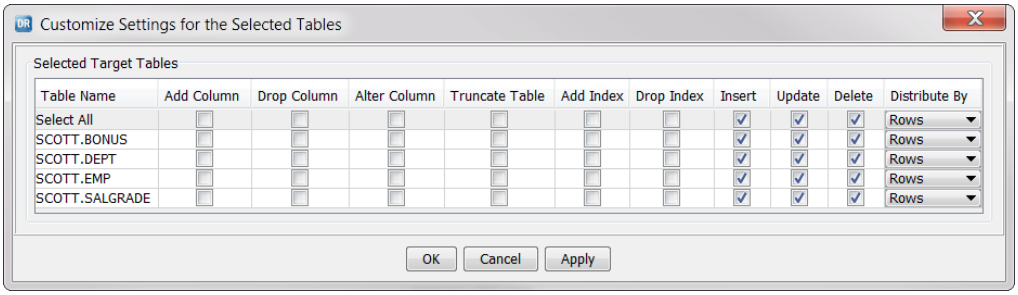
Customizing Apply Settings for Target Tables

You can optionally customize apply settings for individual target tables, provided that the target table has the same name and schema as the source table. You can customize the types of DDL and DML operations to apply and whether to distribute change records by rows or tables.

Note: Data Replication does not replicate DDL operations to Apache Kafka targets.

1. On the **Map Tables** tab, click **Settings**.

   The following image shows the **Customize Settings for the Selected Tables** dialog box:



2. To indicate the types of DDL changes to apply to a table, select the **Add Column**, **Drop Column**, **Alter Column**, **Truncate Table**, **Add Index**, and **Drop Index** check boxes, as needed. By default, these check boxes are not selected.

   Tip: To select a DDL operation type for a single table, select the check box in the row for the table. To select a DDL operation type for all listed tables, select the check box in the **Select All** row.

3. To not apply some types of DML changes, clear the **Insert**, **Update**, and **Delete** check boxes, as needed. By default, these check boxes are selected and Inserts, Updates, and Deletes are applied.

   Tip: To not apply a DML operation type for single table, clear the check box in the row for the table. To not apply a DML operation type for all of the listed tables, clear the check box in the **Select All** row.

4. In the **Distribute by** column, select the type of algorithm that Data Replication uses for distributing Insert, Update, and Delete records to threads for parallel apply processing. Options are:

   - **Rows**. Change records for a table are distributed across threads. This option is the default for Oracle targets.
   - **Tables**. Change records for a table are assigned to a separate thread. This option is the default for DB2, Microsoft SQL Server, and Merge Apply targets. Informatica recommends that you select the **Tables** option when replicating a large number of small tables or when replicating data to an appliance such as Netezza or to a target on AIX, Solaris SPARC, or HP-UX systems that have certain

CPU architecture such as Itanium 64-bit. With this option, Data Replication selects the thread to use for a table and dynamically distributes records as follows:

- In SQL Apply mode, if the apply.subtasks_enabled runtime parameter is set to 1 and the change volume on a thread begins to cause a bottleneck, Data Replication dynamically spawns a subtask to process some of the change records. Data Replication uses the thread with the least load. If necessary, Data Replication can spawn multiple subtasks on multiple subtask threads. To prevent Data Replication from spawning too many threads, set the minimum amount of data, in MB per cycle, that is available to distribute to a subtask before spawning a new thread. To specify this minimum threshold, enter a value for apply.distribute_by_obj_size_threshold parameter on the **Runtime Settings** tab > **Advanced Settings** view. In Merge Apply and Audit Apply modes, Data Replication does not support Applier subtasks.
- If the change volume on a thread is low, other tables that also have low change volumes can share the thread.

This setting is synchronized with the **Distribute by** setting in the **Map Columns** dialog box.

5. Click **OK** to save changes and close the dialog box, or click **Apply** to save changes without closing the dialog box.

Related Topics:

- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "DDL Replication" on page 53
- "DDL Replication Considerations" on page 54
- "Customizing Column Mappings" on page 240
- "CREATE TABLE and ADD COLUMN Operations" on page 55
- "Datatype Conversion Rules" on page 52
- "Replication of TRUNCATE TABLE Operations" on page 57

Configuring the Start Points for Extractor and Applier Tasks

In the Data Replication Console, you can configure the start points for change data extraction and for apply processing.

The Extractor uses the Start Point value to determine the point from which to begin reading the transaction or event logs. The Applier uses the Sync Point value to determine the point from which to begin applying DML changes.

Configuring the Start Point for the Extractor Task

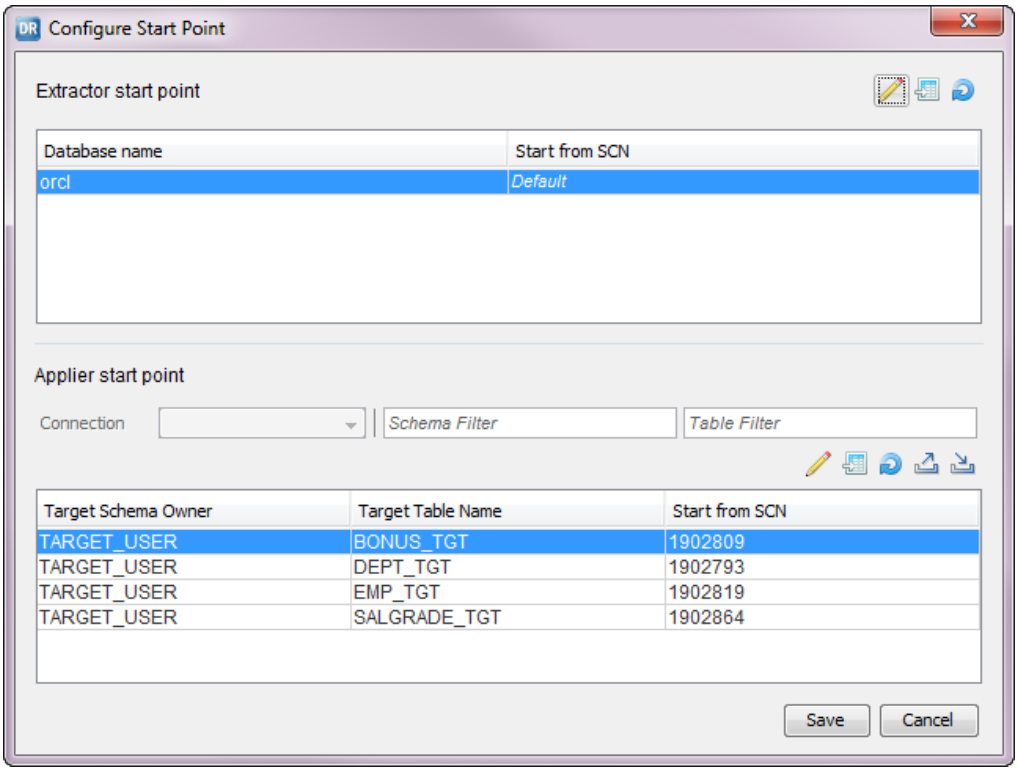
You can specify a Start Point value for a source database or instance to configure the Extractor to start processing the logs that contain the changes from a particular point. The Start Point value that you specify overrides the default Start Point value that the Data Replication Console calculates when you save the configuration.

Note: For Oracle sources, the logs are redo logs. For DB2 and Microsoft SQL Server sources, the logs are transaction logs. For MySQL sources, the logs are binary log files.

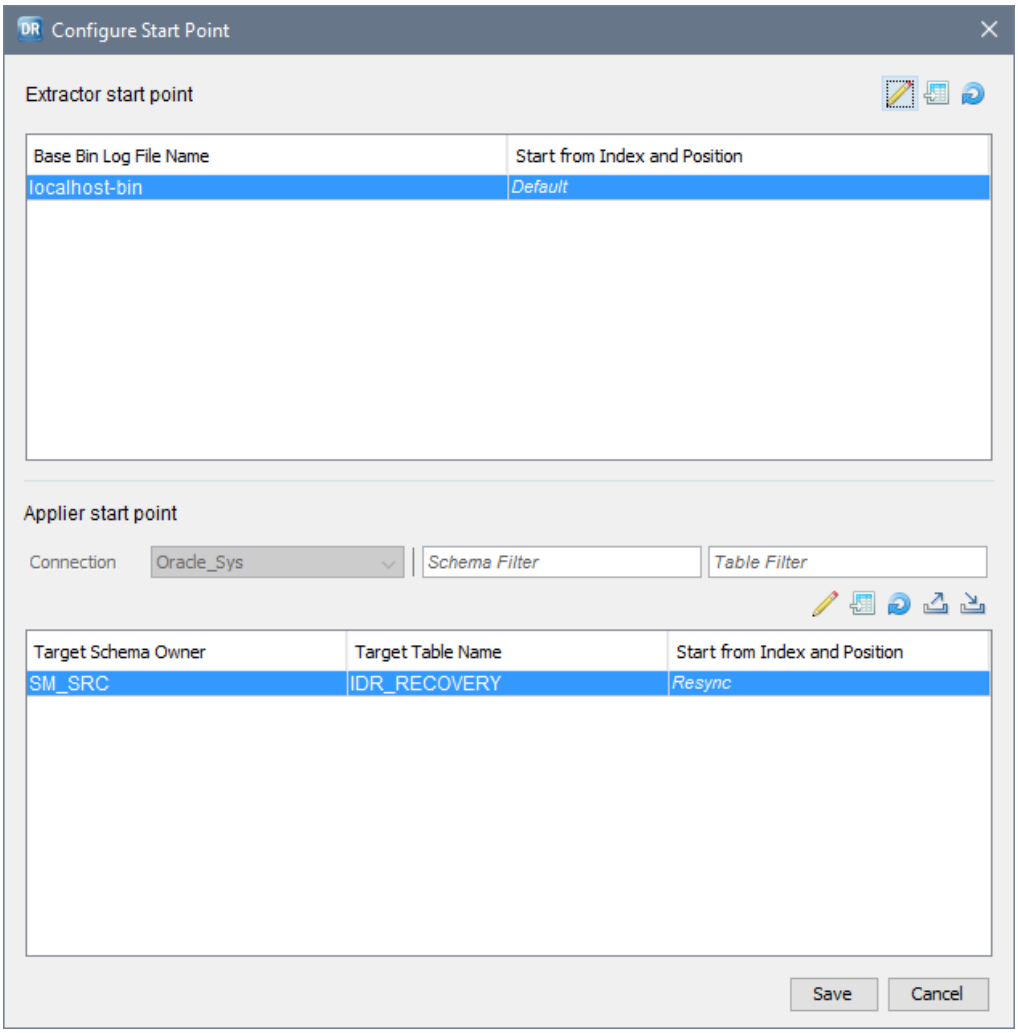
For the Start Point to take effect for Microsoft SQL Server and Oracle sources, ensure the extract.process_old_logs runtime parameter is set to 0.

1. On the **Map Tables** tab, click **Configure Start Point**.

   The **Configure Start Point** dialog box appears. The following image shows this dialog box for an Oracle database:



   The following image shows the **Configure Start Point** dialog box for a MySQL source:



2. Under **Extractor start point**, select the row for which to override the default Start Point value.
3. If you want the Extractor to start processing the log from the current SCN, log coordinates, or LSN value, right-click the row and click **Get SCN from Source**, **Get Index and Position from Source,** or **Get LSN from Source**.

   The current value appears in the **Start from SCN**, **Start from Index and Position,** or **Start from LSN** column.

   Tip: To discard changes and fall back to the last saved SCN, log coordinates, or LSN, click the **Reset** button above the list box, or right-click the row and click **Reset**.
4. If you want the Extractor to start processing the log from a particular SCN, log coordinates, or LSN value, complete the following steps:
   a. Select a row and click the **Edit** icon button.

      The **Edit Extractor SCN** , **Edit Extractor Index and Position**, or **Edit Extractor LSN** dialog box appears.
   b. In the **System change number**, **Index and Position,** or **Log sequence number** field, enter the SCN, log coordinates, or LSN from which the Extractor will start reading the log.

      If you enter -1 or 0, the Extractor starts reading the log from the default Start Point value.

c. Click **Set**.

The specified SCN, log coordinate, or LSN value appears in the **Start from SCN**, **Start from Index and Position,** or **Start from LSN** column.

5. For Microsoft SQL Server sources that have multiple databases listed under **Extractor start point**, repeat steps 2 through 4 for each mapped database.
6. Click **Save**.

Related Topics:

- "Start Point for the Extractor Task" on page 30

Examples of Setting the Start Point

Review the following examples to learn how to set the Start Point value to customize where the Extractor begins processing in the logs.

These examples are based on the following assumptions:

- You have a Microsoft SQL Server source.

  Note: To apply these examples to other source types, use the LSN, log coordinate, or SCN format that is appropriate for the source type.
- The **Extract Range** tab identifies a set of transaction logs with a lowest LSN value of 38000000001600001.
- The extract.process_old_logs runtime parameter is set to 0.
- When InitialSync completed processing, it wrote a Sync LSN value of 38000000035400132 to the configuration. Later, you specified a Start LSN value on the **Map Tables** tab. When the Extractor starts processing the logs, it begins from the Start LSN that you specified.

Example 1. Default LSN

You specify an LSN value of -1 for the Start Point.

The Extractor starts reading the transaction log from the default start LSN.

Note: For all sources, enter -1 to specify the default Start Point value.

Example 2. Start LSN Lower than the Lowest LSN

For the Start Point, you specify an LSN value of 37000000063700250, which is lower than the lowest available LSN of 38000000001600001.

The transaction log that contains the specified start LSN is not available. However, a transaction log that contains higher LSN values is available. In this case, the Extractor ends with the following error message:

```
The transaction log that contains the LSN 37000000063700250 from which the Extractor
must start reading the database logs is not available. The lowest available LSN is
38000000001600001 (database ID 5.)
```

Example 3. Start LSN Greater Than the Current LSN

For the Start Point, you specify an LSN value of 39000000067800137, which is greater than the current LSN of 39000000037800272.

The transaction log that contains the specified start LSN is not yet available. However, a transaction log that contains lower LSN values is available. In this case, the Extractor does not extract any data and ends successfully. When log records with an LSN that is equal to or greater than 39000000067800137 appear in the log, the Extractor extracts these records and writes them to intermediate files.

Example 4. Start LSN Greater Than the Sync LSN

For the Start Point, you specify an LSN value of 38000000039900002, which is greater than the Sync LSN of 38000000035400132.

The Extractor extracts changes starting from the specified start LSN. The Extractor does not extract the changes that occurred after initial synchronization and before the specified start LSN.

Use this scenario to reduce Extractor processing time if the changes that occurred between initial synchronization and the start LSN are not for mapped tables.

Changing the Sync Point for the Applier Task

Change the Sync Point value for one or more mapped target tables if you want the Applier to start applying changes from a particular point in the source logs or if you need to mark tables for InitialSync resynchronization.

The Sync Point value determines the point up to which InitialSync synchronized the target tables with the source tables. When you run the replication, the Applier skips DML changes that have an SCN, log coordinate, or LSN value that is lower than the Sync Point value.

1. On the **Map Tables** tab, click **Configure Start Point**.

   The **Configure Start Point** dialog box appears.

2. Under **Applier start point**, select a target connection in the **Connection** list.

   Select the connection to the target database that contains the table or tables for which you want to change the Sync Point value.

3. Select one or more target tables for which to change the Sync Point value in one of the following ways:

   - To filter the tables by schema owner, enter a filter pattern in the **Schema Filter** field. You can use the question mark (?) and asterisk (*) wildcards. For case-sensitive filtering, enclose the filter value in double quotation marks.
   - To filter the tables by table name, enter a filter pattern in the **Table Filter** field. You can use the question mark (?) and asterisk (*) wildcards. For case-sensitive filtering, enclose the filter value in double quotation marks.
   - To select the tables individually, hold down the CTRL key and click each table row.
   - To select all of the mapped tables, either press CTRL + A or right-click any table row and click **Select All**.

   Tip: To clear all selections, right-click any table row and click **Select None**. To clear all selections and select the tables that were not previously selected, right-click any table row and click **Select Inverse**.

4. If you need to resynchronize any target tables by using InitialSync, complete the following steps:

   a. Select the target tables that you want to resynchronize and click the **Edit** icon button.

      The **Edit Applier SCN**, **Edit Applier Index and Position**, or **Edit Applier LSN** dialog box appears.

   b. Click the **Resync** button.

      The Data Replication Console displays *Resync* in the **Start from SCN**, **Start from Index and Position**, or **Start from LSN** column for each selected table.

   Data Replication sets the Sync Point values for these tables to -1 in the configuration. If you previously saved the configuration, the Data Replication Console prompts you to synchronize these tables when you save the configuration again.

5. If you want the Applier to start applying change data from the current SCN, log coordinates, or LSN, select one or more target table rows. Then right-click the selection and click **Get SCN From Source**, **Get Index and Position from Source**, or **Get LSN From Source**.

   The current SCN, log coordinates, or LSN values appear in the **Start from SCN**, **Start from Index and Position**, or **Start from LSN** field for each selected table.
6. If you want the Applier to start applying change data from a particular SCN, log coordinates, or LSN value, complete the following steps:
   a. Select one or more target table rows and click the **Edit** icon button.

      The **Edit Applier SCN**, **Edit Applier Index and Position,** or **Edit Applier LSN** dialog box appears.
   b. Enter a valid SCN, log coordinates, or LSN value that you want to use as the Sync Point. For a MySQL binary log file, use the format *log_file_numeric_suffix*:*log_position*.
   c. Click **Set**.

      The specified SCN, log coordinate, or LSN value appears in the **Start from SCN**, **Start from Index and Position**, or **Start from LSN** column for all of the selected tables.
7. For configurations that have multiple targets, repeat steps 2 through 6, as needed, for each target connection.

   Tip: To undo changes that you made in steps 4, 5, or 6, click the **Reset** icon button.
8. Click **Save**.

Related Topics:

- "Sync Point Value" on page 272
- "InitialSync Handling of Target Table Constraints" on page 273

Exporting Sync Point Values to a .CSV File

If you need to override the Sync Point values in the configuration for multiple target tables and do not want to do it by selecting each target table, you can export the Sync Point values to a .csv file and then override the Sync Point values in the file. After saving the .csv file, import the file into the configuration so that the Applier can use the override values to determine the point at which to start applying changes.

1. On the **Map Tables** tab, click **Configure Start Point**.

   The **Configure Start Point** dialog box appears.
2. Under **Applier start point**, click the **Export** icon.
3. Select the system to which to export the list of Sync Point values. Options are:
   - **Main server**. Export Sync Point values to a .csv file on the system where the Server Manager Main server runs.
   - **Local**. Export Sync Point values to a .csv file on the system where the Data Replication Console runs.
4. In the **Save** dialog box, enter a file name and location.
5. Click **Save**.

   The Data Replication Console exports Sync Point values to the specified .csv file.
6. Click **OK**.

Sync Point .CSV File

The .csv file to which you export Sync Point values for mapped target tables has a specific format.

File Format

The file uses the following format:

```
#tgt name: target_connection_name
#id: DbConnMapID
object_ID, Sync_Point # "source_schema"."source_table"
```

The following table describes the variables in this file format:

Variable	Description
target_connection_name	The name of the target connection. For configurations that have multiple targets, this file contains a section with Sync Point values for each target connection.
DbConnMapID	The internal unique identifier for the combination of the configuration and target connection.
object_ID	The unique identifier of the source table.
Sync_Point	The Sync Point value of the source table that is identified by the object ID. The Applier uses the Sync Point value to start applying changes to the mapped target table from a particular point in the transaction log.
source_schema	The name of the source schema.
source_table	The name of the source table.

Sample File

The following sample file contains Sync Point values for a configuration that has two targets:

```
#tgt name: ora_tgt_1
#id: 6
90222, 169192565862 # "SRC"."TABLE1"
90229, 169192565863 # "SRC"."TABLE2"
#tgt name: ora_tgt_2
#id: 7
90222, 169192565904 # "SRC"."TABLE1"
90229, 169192565905 # "SRC"."TABLE2"
```

Importing Sync Point Values from a .CSV File

You can import Sync Point values for mapped target tables from a .csv file into a configuration. The Sync Point values are then available to the Applier for target apply processing. You must have previously exported the Sync Point values for one or more target tables to a .csv file from the Data Replication Console or with the export scn command from the Server Manager CLI.

To import Sync Point values for selected source tables, edit the .csv file to remove the lines for the tables for which you do not want to import Sync Point values.

Important: Do not import Sync Point values that were exported from another configuration.

1. On the **Map Tables** tab, click **Configure Start Point**.

   The **Configure Start Point** dialog box appears.

2. Under **Applier start point**, click the **Import** icon.

3. Select the system that contains the .csv file with the exported Sync Point values. Options are:
   - **Main server**. Import Sync Point values from a .csv file on the system where the Server Manager Main server runs.
   - **Local**. Import Sync Point values from a .csv file on the system where the Data Replication Console runs.
4. In the **Open** dialog box, enter the .csv file name and location.
5. Click **Open**.

   The Data Replication Console imports Sync Point values from the selected .csv file.
6. Click **OK**.

Configuring Conflict Resolution

If you replicate change data to a target database on which other change data activity occurs, you can define conflict resolution rules for the target tables to handle conflicts that might occur during Apply processing.

Configure conflict resolution rules if you perform bidirectional replication or replication from multiple sources to a single target in SQL Apply mode. You cannot configure conflict resolution rules for mapped tables that are replicated in Merge Apply or Audit Apply mode.

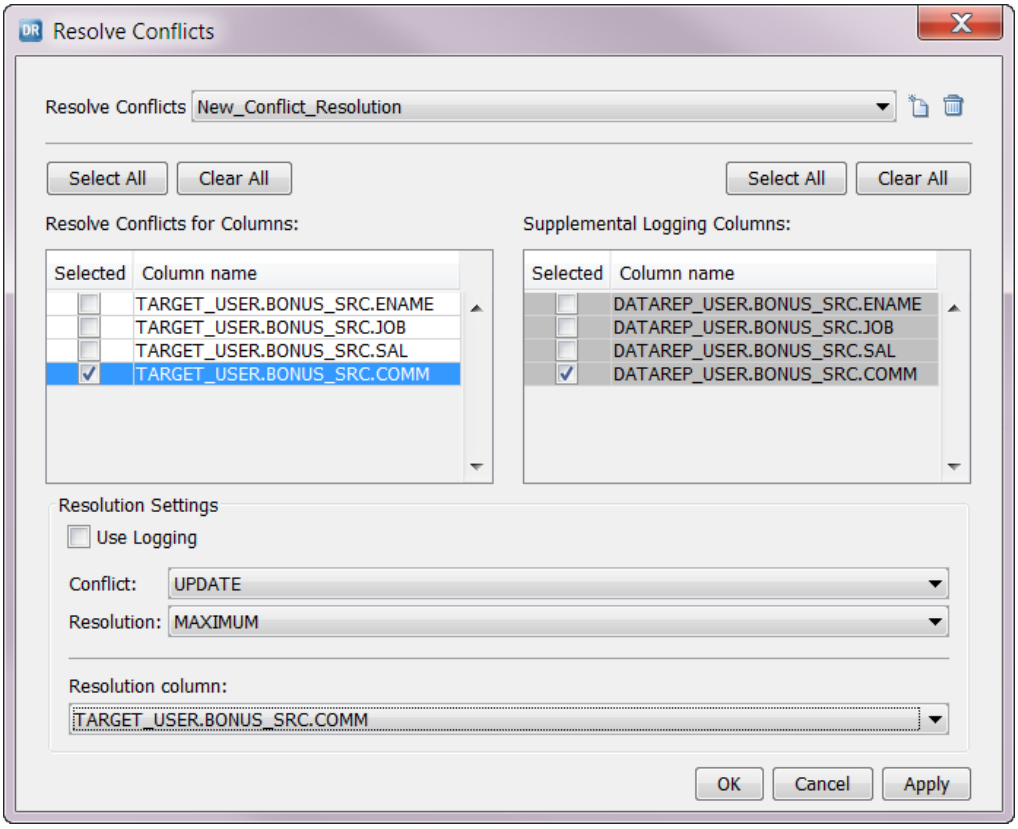
1. On the **Map Tables** tab, select the target table for which to define a conflict resolution rule.
2. Click the **Resolve Conflicts** icon button above the source tables filter field, or right-click the mapped table on the target and select **Resolve Conflicts**.

   The **Resolve Conflicts** dialog box appears.
3. To define a conflict resolution rule, click the **New Conflict Resolution** button to the right of the **Resolve Conflicts** field.

   A dialog box displays the default conflict resolution name New_Conflict_Resolution.
4. In the **Conflict resolution name** field, edit the default conflict resolution rule name. Then click **OK**.

   Note: Conflict resolution rule names can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates conflict resolution rule names that are longer than 100 characters.

The following image shows the **Resolve Conflicts** dialog box with the default conflict resolution name in the **Resolve Conflicts** field:



5. In the **Conflict** list, select the conflict type for which you are defining the rule.

   The following table describes the types of conflicts that Data Replication handles:

Conflict Type	Description
Delete	Occurs when a row that is referenced in an SQL Delete operation does not exist in the target table.
Uniqueness	Occurs when the Applier applies change data that violates a uniqueness integrity constraint such as primary key or unique index.
Update	Occurs when the row that is referenced in an SQL Update operation has different values on the source and target. When you configure a replication job, you must select the columns for which Data Replication compares source and target values. For these columns, the Data Replication Console enables supplemental logging so that the Applier can get the before-image and after-image values to compare the source and target rows.

6. In the **Resolution** list, select a resolution strategy.

The following table describes the conflict resolution strategies that are available by conflict type:

Conflict Resolution Strategy	Conflict Types	Description
Custom	All	Calls a stored procedure that you previously defined to handle conflicts. You can define a stored procedure by using commands that the SQL Script Engine provides. When an SQL operation on the target table causes conflict, the Applier calls the stored procedure and passes operation metadata and the before and after values for all of the columns for which supplemental logging is enabled as procedure parameters. These parameters are: - **OP_XID.** Transaction ID. - **OP_CODE.** A code for the operation type, which can be D for deletes, I for inserts, and U for updates. - **OP_TIME.** The date and time of the operation. - **OP_CMT_SCN.** The SCN value for the operation commit. - **OP_CMT_TIME.** The commit time for the operation. - **OP_NUM_IN_TX.** The sequence number of the operation in the transaction. - ***column_name*_OLD, *column_name*_NEW.** The before and after values for the column <*column_name*>. These parameters repeat for each column for which you enabled additional logging. For more information about the SQL Script Engine commands, see the *Informatica Data Replication Scripting Guide*.
Discard	All	Discards the source values and retains the existing values in the target table.
Maximum, Minimum	Update	For these resolution strategies, you must select a resolution column. For conflicting rows, Data Replication compares source and target values in the resolution column. - For the Maximum strategy, if the target value is greater than the source value, Data Replication uses the Discard strategy. Otherwise, Data Replication uses the Overwrite strategy. - For the Minimum strategy, if the target value is less than the source value, Data Replication uses the Discard strategy. Otherwise, Data Replication uses the Overwrite strategy.
Overwrite	- Update - Uniqueness	Overwrites records in the target table with the values from the source table.

7. For the UPDATE conflict type, in the **Resolve Conflicts for Columns** box, select the columns for which the Applier compares the source and target values to detect conflicts.

   Tip: To select or clear all of the columns, click **Select All** or **Clear All** on the left side of the dialog box.

   The Data Replication Console enables supplemental logging for the selected columns.

8. For the CUSTOM resolution strategy, create a stored procedure that the Applier uses to handle conflict resolution operations:

   a. In the **Supplemental Logging Columns** box, select the columns for which to enable supplemental logging and make the before and after images available to the stored procedure.

      Tip: To select or clear all of the columns for supplemental logging, click **Select All** or **Clear All** on the right side of the dialog box.

b. Click **Create Procedure**. Then enter the procedure name and click **OK**.

   Use the following naming pattern for procedure names:

   - For DB2 and Oracle targets, use "*schema_name*"."*procedure_name*".
   - For Microsoft SQL Server targets, use *owner_name*.*procedure_name*.

   For Oracle targets, the procedure name cannot exceed 30 characters. For DB2 and Microsoft SQL Server targets, the procedure name cannot exceed 128 characters.

c. In the **Generate Procedure** dialog box, create the stored procedure.

   The Data Replication Console provides a stored procedure template for the selected conflict type. Add information in the procedure body.

d. Click **Execute** to run the generated procedure script immediately, or click **Save** to save the procedure to a file on the file system so that you can run the procedure later manually.

9. For the MAXIMUM and MINIMUM resolution strategies, select a resolution column in the **Resolution column** list.
10. To enable logging for conflicts of the specified type, select **Use logging**.

    The Applier logs information about these conflicts to the log file that is specified in the **Conflict Logging** box on the **Runtime Settings** tab > **General** view.
11. Click **OK** to save the rule.
12. To handle Update conflicts for Microsoft SQL Server 2008 R2 sources, verify that the extract.mssql.process_updates_as_updates parameter is set to the default setting of 1 on the **Runtime Settings** tab > **Advanced Settings** view.

Related Topics:

- "Conflict Resolution for Replicated Data" on page 41

Example of Configuring a MAXIMUM Resolution Strategy for Update Conflicts

This example demonstrates how you can use a MAXIMUM resolution strategy to resolve Update conflicts.

The example uses a table that contains a service column with the name of MODIFIED. This column holds the date and time of Insert or Update operations for each row. When an Update conflict occurs between the source and target values, the Applier uses this column to determine the latest changes. If changes on the target occurred later than the changes on the source, Data Replication retains the target values. Otherwise, Data Replication overwrites the target values with the source values.

1. Create the table CONFLICTS_DEMO on the source.

   Use the following statement:

```
CREATE TABLE CONFLICTS_DEMO
(
    ID INTEGER PRIMARY KEY NOT NULL,
    NAME VARCHAR2(20),
    MODIFIED DATE
);
```

2. Create a trigger on the source and target that sets a timestamp in the MODIFIED column when an Update or Insert occurs.

   Use the following statement:

```
CREATE OR REPLACE TRIGGER UPDATE_DATE
BEFORE INSERT OR UPDATE OF ID, NAME ON CONFLICTS_DEMO
```

```
FOR EACH ROW
BEGIN
  :NEW.MODIFIED := sysdate;
END;
```

3. Insert a row into the CONFLICTS_DEMO table on the source.

   Use the following statement:

   ```
   INSERT INTO CONFLICTS_DEMO (ID, NAME) VALUES (1, 'name1');
   ```

   Oracle inserts the row and executes the trigger to get a value for the MODIFIED column.

4. In the Data Replication Console, connect to the source and target databases. Then generate a target table named CONFLICTS_DEMO based on the source table schema.

5. Map the CONFLICTS_DEMO source table to the CONFLICTS_DEMO target table. Configure other replication settings.

   Note: You must select the **SQL Apply** option to map target tables for which you plan to configure conflict resolution. The Data Replication Console does not allow you to configure conflict resolution for tables that are mapped with the **Audit Apply** or **Merge Apply** option.

6. On the **Map Tables** tab, select the CONFLICTS_DEMO target table.

7. Click the **Conflict Resolution** icon button above the source tables filter field, or right-click the table row and click **Conflict Resolution**.

   The **Conflict Resolution** dialog box appears.

8. Click the **New Conflict Resolution** button next to the **Conflict Resolution** list. Then enter a name for the conflict resolution rule and click **OK**.

   For example, enter max_date_update as the rule name.

9. In the **Resolution Settings** box, select **UPDATE** from the **Conflict** list.

10. Under **Resolve Conflicts for Columns**, select the NAME column.

    When you save the configuration, the Data Replication Console enables supplemental logging for this column.
    When you run the Applier, it searches for the target row to update by using the primary key column and the NAME column. If the NAME column value was also updated in the corresponding row on the target, the Applier cannot find the row and attempts to resolve the conflict.

11. In the **Resolution Settings** box, select the **MAXIMUM** resolution strategy from the **Resolution** list.

12. Select the MODIFIED column in the **Resolution column** list.

    When you save the configuration, the Data Replication Console enables supplemental logging for this column. The Applier uses the before values for this column to determine a maximum value between the source and target values. If changes on the target occurred later than on the source, the Applier retains the target values. Otherwise, the Applier overwrites the target values with the source values.

13. Enable logging to get debug information and the before and after images for the rows in conflict.

    a. Select the **Use logging** option.

    b. On the **Runtime Settings** tab > **General** view, specify a location for the log file in the **Conflict Logging** box.

14. Save the configuration.

15. To introduce an Update conflict to test the configuration, complete the following steps:

    a. Run the replication to replicate an Insert operation.

    b. Update the inserted row on the source and target with different values to create an Update conflict.

Use the following statement on the source:

```
UPDATE CONFLICTS_DEMO SET NAME='name2' WHERE ID=1;
```

Use the following statement on the target:

```
UPDATE CONFLICTS_DEMO SET NAME='name3' WHERE ID=1;
```

16. Run the replication again.

    The Applier detects the conflict and applies the latest changes. You can view conflict details in the log file that you specified on the **Runtime Settings** tab > **General** view.

Example of Configuring Custom Conflict Resolution

This example demonstrates how to configure a custom resolution strategy for Update conflicts. The example also creates a custom procedure that adds information about the operations that cause Update conflicts and the before and after images of columns for which supplemental logging is enabled to a log table on the target.

In this example, Oracle is both the source and target type.

1. Create a table on the source for which to configure conflict resolution.

   Use the following statement:

   ```
   CREATE TABLE CONFLICTS_DEMO
   (
       ID INTEGER PRIMARY KEY NOT NULL
       NAME VARCHAR2(20),
       MODIFIED DATE
   );
   ```

2. Insert a row to the CONFLICTS_DEMO table.

   Use the following statement:

   ```
   INSERT INTO CONFLICTS_DEMO (ID, NAME, MODIFIED) VALUES (1, 'name1', sysdate);
   ```

3. In the Data Replication Console, connect to the source and target databases. Then generate the target table named CONFLICTS_DEMO based on the source table schema.
4. Map the CONFLICTS_DEMO source table to the CONFLICTS_DEMO target table. Configure other replication settings.

   Note: You must select the **SQL Apply** option to map target tables for which you plan to configure conflict resolution. The Data Replication Console does not allow you to configure conflict resolution for tables that are mapped with the **Audit Apply** or **Merge Apply** option.
5. On the **Map Tables** tab, select the CONFLICTS_DEMO target table.
6. Click the **Conflict Resolution** toolbar button, or right-click the table row and click **Conflict Resolution**.

   The **Conflict Resolution** dialog box appears.
7. Click the **New Conflict Resolution** button next to the **Conflict Resolution** list. Then enter a name for the conflict resolution rule and click **OK**.

   For example, enter custom_update as the rule name.
8. In the **Resolution Settings** box, select **UPDATE** from the **Conflict** list.
9. Under **Resolve Conflicts for Columns**, select the NAME column.

   When you save the configuration, the Data Replication Console enables supplemental logging for this column.
   When you run the Applier, it searches for the target row to update by using the primary key column and NAME column. If the NAME column value was also updated in the corresponding row on the target, the Applier cannot find the row and attempts to resolve the conflict.

10. In the **Resolution Settings** box, select the **CUSTOM** resolution strategy from the **Resolution** list.
11. Under **Expression**, click the **Create Procedure** button. Then enter a name for the custom procedure and click **OK**.

    For example, use *schema_name*.CONFLICTS_DEMO_PROC as procedure name.
    The Data Replication Console opens the **Generate Procedure** dialog box and provides a template for the custom procedure. The operation metadata and the before and after values for the columns that you selected for supplemental logging are available as procedure parameters.
12. Create a log table on the target for logging information about the operations that cause Update conflicts.

    Use the following statement to create the log table:

```
CREATE TABLE CONFLICTS_DEMO_LOG
(
    OP_XID VARCHAR2(20),
    OP_CODE CHAR(1),
    OP_TIME DATE,
    OP_CMT_SCN VARCHAR2(20),
    OP_CMT_TIME DATE,
    OP_NUM_IN_TX NUMBER(20),
    NAME_OLD VARCHAR2(20),
    NAME_NEW VARCHAR2(20)
);
```

13. Complete the CREATE PROCEDURE statement.

    For example, the following procedure adds information about the operations that cause Update conflicts and the before and after images of the columns for which you enabled supplemental logging to the previously created log table:

```
CREATE OR REPLACE PROCEDURE DEMO_TARGET.CONFLICTS_DEMO_PROC
 (
     OP_XID_PAR VARCHAR2,
     OP_CODE_PAR CHAR,
     OP_TIME_PAR DATE,
     OP_CMT_SCN_PAR VARCHAR2,
     OP_CMT_TIME_PAR DATE,
     OP_NUM_IN_TX_PAR NUMBER,
     NAME_OLD_PAR DEMO_TARGET.CONFLICTS_DEMO.NAME%TYPE,
     NAME_NEW_PAR DEMO_TARGET.CONFLICTS_DEMO.NAME%TYPE
 )
 IS
 BEGIN
   INSERT INTO DEMO_TARGET.CONFLICTS_DEMO_LOG
   (
    OP_XID,
       OP_CODE,
       OP_TIME,
       OP_CMT_SCN,
       OP_CMT_TIME,
       OP_NUM_IN_TX,
       NAME_OLD,
       NAME_NEW
   )
   VALUES
   (
       OP_XID_PAR,
       OP_CODE_PAR,
       OP_TIME_PAR,
       OP_CMT_SCN_PAR,
       OP_CMT_TIME_PAR,
       OP_NUM_IN_TX_PAR,
       NAME_OLD_PAR,
       NAME_NEW_PAR
   );
   COMMIT;
 END;
```

Note: The example uses names for the procedure parameters other than the default names because the log table uses these names as column names.

14. Click **Save** and save the generated procedure script to a temporary location.
15. Execute the generated script on the target.
16. Enable logging to get debug information and the before and after images for the rows in conflict:
    a. Select the **Use logging** option.
    b. On the **Runtime Settings** tab > **General** view, specify a location for the log file in the **Conflict Logging** box.
17. Save the configuration.
18. To introduce an Update conflict to test the configuration, complete the following steps:
    a. Run the replication to replicate an Insert operation.
    b. Update the inserted row on the source and target with different values to create an Update conflict.

       Use the following statement on the source:

       ```
       UPDATE CONFLICTS_DEMO SET NAME='name2' WHERE ID=1;
       ```

       Use the following statement on the target:

       ```
       UPDATE CONFLICTS_DEMO SET NAME='name3' WHERE ID=1;
       ```
19. Run the replication again.

    The Applier detects the conflict and adds a row to the CONFLICTS_DEMO_LOG table. You can review this table later to manually resolve the conflict that occurred.

    You can view conflict details in the log file that you specified on the **Runtime Settings** tab > **General** view.

Customizing Column Mappings

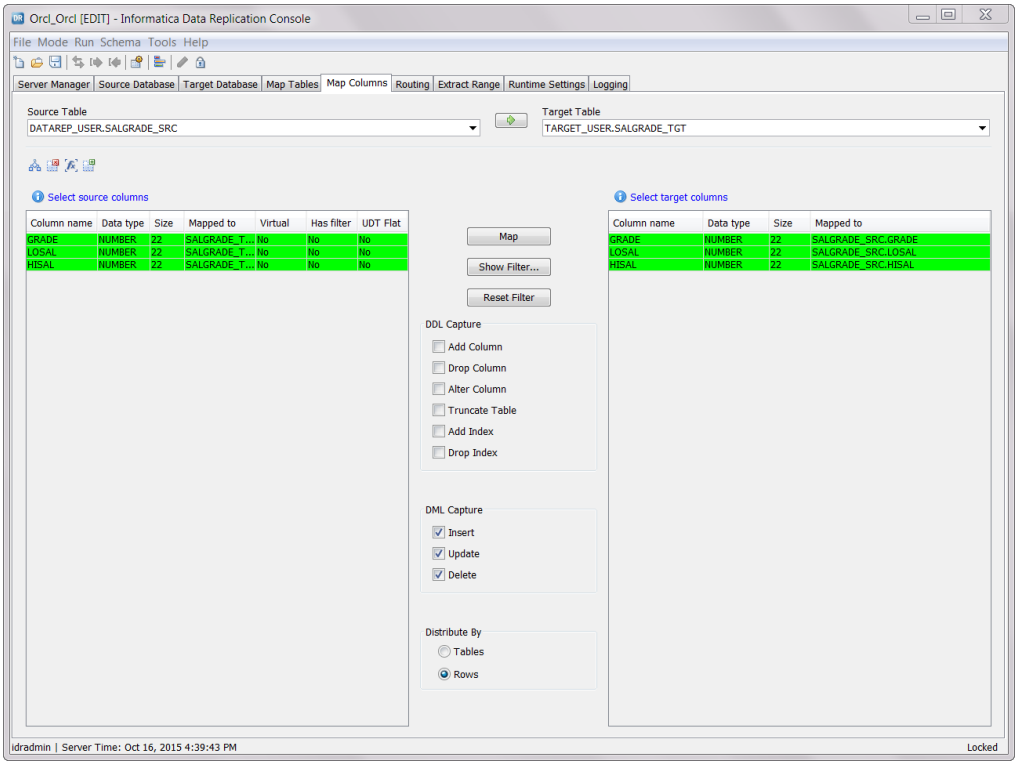
When you map tables, Data Replication maps the source columns to the target columns that have matching names by default. On the **Map Columns** tab, you can manually map or unmap columns. Also, you can set some capture options for a pair of mapped tables.

Note: For each mapped table, you must define one or more column mappings. If a configuration contains mapped tables for which you did not define any column mappings, you cannot save the configuration. Map

only columns that have a supported column type. For more information, see the *Informatica Data Replication Datatype Mapping Reference*.

1. Click the **Map Columns** tab.

   The following image shows this tab:



2. In the **Source Table** list, select the table that contains the columns for which you want to change mappings or set capture options.

   The corresponding target table appears in the **Target Table** list.

3. To remove a column mapping, use one of the following procedures:

   - Double-click the source or target row and click **Yes** in the confirmation box.
   - Right-click the column row and click **Remove Mapping**. Then click **Yes** in the confirmation box.

4. To remap columns, perform the following steps:

   a. Select the source column row.

      Tip: To undo your selection, right-click the row and select **Clear Selection**.

   b. Select the target column row.

   c. Click **Map**.

5. Under **DDL Capture**, select the check boxes for the types of DDL changes that you want to capture for the mapped table.

   Note: The **DDL Capture** check boxes are not available if the configuration has an Apache Kafka target. DDL replication is not supported for Kafka targets.

6. Under **DML Capture**, clear one or more of the following check boxes to filter the types of DML changes to capture: **Insert**, **Update**, and **Delete**.

   By default, all check boxes are selected and all types of DML changes are captured.

7. Under **Distribute By**, select the type of algorithm that Data Replication uses for distributing Insert, Update, and Delete records to threads for parallel apply processing. Options are:

   - **Rows**. Change records for a table are distributed across threads. This option is the default for Oracle sources.
   - **Tables**. Change records for a table are assigned to a main thread. This option is the default for DB2 and Microsoft SQL Server sources. Informatica recommends that you select the **Tables** option only when replicating many small tables or when replicating data to an appliance such as Netezza or to a target on AIX, Solaris SPARC, or HP-UX systems with certain CPU architecture such as Itanium 64-bit. With this option, Data Replication selects the thread to use for a table and dynamically distributes records as follows:
     - If the change volume on the main thread begins to cause a bottleneck, Data Replication dynamically spawns a subtask on another thread to process some of the change records. Data Replication uses the thread with the least load. If required, Data Replication can spawn multiple subtasks on multiple subtask threads. To prevent Data Replication from spawning too many threads, set the minimum amount of data, in MB per cycle, that is available to distribute to a subtask before spawning another thread. To specify this minimum threshold, enter a value for the apply.distribute_by_obj_size_threshold parameter on the **Runtime Settings** tab > **Advanced Settings** view.
     - If the change volume on the main thread is low, other tables with low change volumes can share the thread.

   This setting synchronizes with the similar setting that you specified in the **Map Tables** > **Settings** dialog box.

   Note: These options are not available if the configuration has an Apache Kafka target.
8. Repeat steps 2 through 7, as needed, for other mapped tables.

Related Topics:

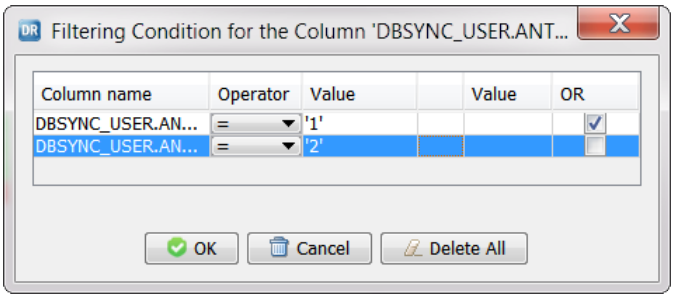
- "Enabling Replication of DDL Changes at the Schema and Table Levels" on page 223
- "Customizing Apply Settings for Target Tables" on page 225

Filtering Column Data

To filter source data for a replication job, you can define filter conditions for one or more source columns. The Data Replication Console then generates a table WHERE clause based on those conditions. The Extractor uses the WHERE clauses to filter the source data from redo logs for replication.

1. On the **Map Columns** tab, select a column for which to create a filter from the source columns list.
2. Click the **Set WHERE Clause** icon button above the source columns box, or right-click a source column row and click **Set WHERE Clause**.

   The **Filtering Condition for the Column 'table.column_name'** dialog box appears.



3. In the **Operator** field, select one of the following comparison operators: =, <>, <, >, >=, <=, IS NULL, IS NOT NULL, LIKE, BETWEEN, or NOT BETWEEN.

   The operators that are available depend on the column datatype.

4. In the **Value** field, enter a value to use with the operator.
5. Select the **OR** check box to add another condition for the current column.

   With the OR connector, at least one of the conditions must be true to filter the data for the column.

   Note: If you clear the **OR** check box, the Data Replication Console removes any conditions that follow the first one.

6. Click **OK**.

   To enable supplemental logging for a condition column, the Data Replication Console creates a virtual index for the table, if a virtual index that includes the condition column does not already exist. To name the index, the Data Replication Console uses the table name followed by the _WHERE suffix.

7. To verify the resulting WHERE filter for the column, click the **Map Columns** tab > **Show Filter** button.

   Tip: To remove the WHERE filter, click **Reset Filter**.

8. Repeat steps 1 through 7 to add filter conditions for other columns.

   Note: Filter conditions for all of the columns must be met to filter data.

The Extractor extracts the records that match the filtering conditions that you defined from the **Map Columns** tab. If you add filters to the configuration after the Extractor processes the database logs, Data Replication does not apply these filters to the replicated data.

Adding Tcl and SQL Expressions

You can add Tcl and SQL expressions to mapped source tables to customize InitialSync and Applier processing.

In the Data Replication Console, define a Tcl or SQL expression for a mapped source table and assign this expression to a virtual column. Then map the virtual column to a column in the target table. You can also copy virtual columns to other tables in the same configuration to reuse Tcl and SQL expressions.

Restrictions:

- Data Replication does not support SQL expressions for targets that use Audit Apply mode, such as Apache Kafka, Cloudera, Hortonworks, and Flat File targets.
- The Tcl Script Engine and SQL Script Engine do not support LOB data. Do not define Tcl scripts or SQL expressions that process LOB data.

Adding Tcl Expressions

After you write a Tcl script based on the information in the *Informatica Data Replication Scripting Guide*, enter the script for a source table in the Data Replication Console.

Before you begin, make sure that the Console is in Edit mode.

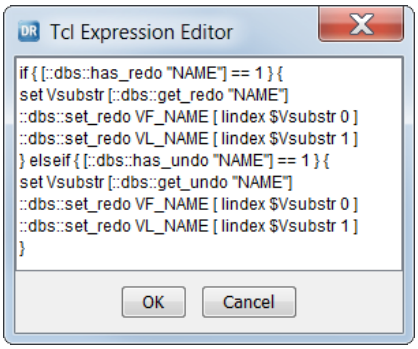
1. On the **Map Tables** tab, select the mapped source table row for which to define a Tcl expression.
2. Click the **Add or Show Table Expression** icon button above the source tables filter field, or right-click the table row and click **Add or Show Table Expression**.

   The **Add/Show expressions** dialog box appears.

3. Click **New**.
4. Enter an expression name and click **OK**.

   An expression name can be no longer than 50 characters and can include only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.
5. In the **Language** field, select **TCL**.
6. Double-click the **Expression** field or click **Edit**.

   The **Tcl Expression Editor** dialog box appears.



7. Enter a Tcl expression for processing the replicated data and click **OK**.

   The Tcl expression appears in the **Add/Show expressions** dialog box.

   Tips:

   - To edit an expression, double-click in the **Expression** field, or select the expression row and click **Edit**.
   - To rename an expression, double-click in the **Name** field. Then edit the expression name and press **Enter**.
   - To delete an expression that is not assigned to any virtual column, select the expression row and click **Delete**.
8. When done, click **Close**.

Next, you can assign the Tcl expression to a virtual column.

Adding SQL Expressions

In the Data Replication Console, you can use data access operators and SQL comparison and logical operators to quickly define an SQL expression.

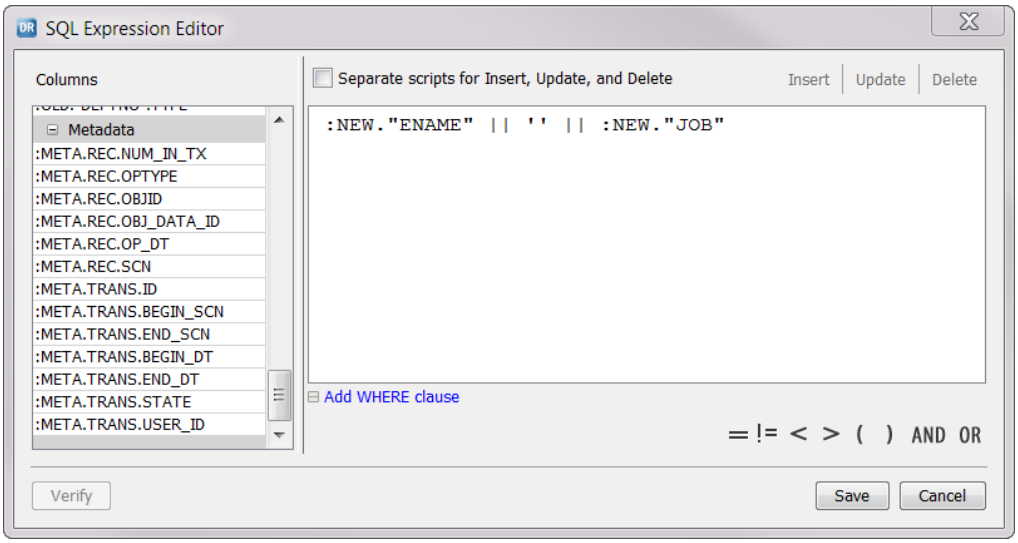
Before you begin, make sure that the Console is in Edit mode.

1. On the **Map Tables** tab, select the mapped source table row for which to define an SQL expression.
2. Click the **Add/Show Table Expression** icon button above the source tables filter field, or right-click the table row and click **Add/Show Table Expression**.

   The **Add/Show expressions** dialog box appears.
3. Click **New**.
4. Enter an expression name and click **OK**.

   An expression name can be no longer than 50 characters and can include only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.
5. In the **Language** field, select **SQL**.
6. Double-click the **Expression** field, or click the **Edit** button.

The **SQL Expression Editor** dialog box appears.



7. To define separate SQL expressions for Insert, Update, and Delete operations, select the **Separate scripts for Insert, Update, and Delete** check box.

   To define a single SQL expression that affects all of these SQL operation types, do not select this check box.

8. Enter an SQL expression in the text box.

   If you selected the **Separate scripts for Insert, Update, and Delete** check box, use the **Insert**, **Update**, and **Delete** buttons to switch from one SQL operation type to another.

   To quickly create an SQL expression, you can use the data access operators in the **Columns** list box and the SQL operators in the bottom right corner. The Columns section of the **Columns** list contains columns in the mapped source table but does not contain virtual columns. Virtual columns cannot be included in an SQL expression. To insert a data access operator, click the operator row. To insert an SQL operator, click the button for the operator.

   Note: Verify that all single and double quotation marks in the SQL expression have the appropriate closing quotation marks. If you save a replication configuration that includes an SQL expression in which one or more closing quotation marks are missing, the Data Replication Console will incorrectly display this SQL expression when you open the configuration later.

9. For Updates and Deletes, if you want to map the virtual column to a primary key column on the target, click **Add WHERE clause**. Then under **WHERE clause**, enter a WHERE clause.

10. Click **Save**.

    The SQL expression appears in the **Add/Show Expression** dialog box.

    Tips:

    - To edit an expression, double-click in the **Expression** field, or select the expression row and click **Edit**.
    - To rename an expression, double-click in the **Name** field. Then edit the expression name and press **Enter**.
    - To delete an expression that is not assigned to any virtual column, select the expression and click **Delete**.

11. When you are finished, click **Close**.

After you assign the defined expression to a virtual column and map the virtual column to a target column, you can open the **SQL Expression Editor** dialog box again and click **Verify** to validate the SQL expression.

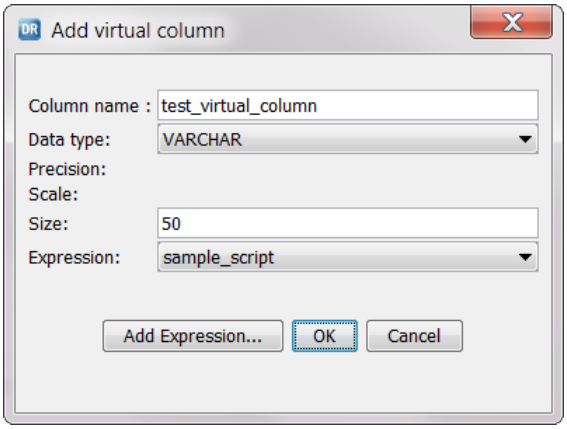
Adding a Virtual Column with an SQL or Tcl Expression

After you define an SQL or Tcl expression for a source table, you must assign it to a virtual column. Then map the virtual source column to a column in the corresponding target table.

Important: Data Replication validates that the virtual column and target column have compatible datatypes. If the datatypes are not compatible, the Data Replication Console displays an error message that prompts you to map the virtual column to a target column that has a compatible datatype. For more information about compatible datatypes, see the *Informatica Data Replication Datatype Mapping Reference*.

1. On the **Map Columns** tab, in the **Source Table** list, select the source table for which you previously defined a Tcl or SQL expression.
2. Click the **Add Virtual Column** icon button above the source columns box, or right-click any source column row and click **Add Virtual Column**.

   The **Add virtual column** dialog box appears.



3. In the **Column name** field, enter a unique virtual column name.

   A virtual column name can be no longer than 50 characters and can include only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.

   Note: Do not create a virtual column that has the same name as another virtual column in the same table.
4. In the **Data type** list, select the virtual column datatype.

   The following table describes the virtual column datatypes:

Datatype	Description
BIGINT	Big integers. Storage size: 8 bytes Range of values: -2^63 (-9223372036854775808) through 2^63-1 (9223372036854775807)
DATE	Combined date and time value. Date format for Tcl scripts: yyyy-mm-dd hh:mm:ss Date format for SQL expressions: target database format

Datatype	Description
DECIMAL(*p*,*s*)	Decimal numbers with the declared precision and scale. Scale must be less than or equal to precision. Valid precision values: 1 through 38 Valid scale values: 0 through 38
DOUBLE	Double-precision floating-point numbers. Storage size: 8 bytes Range of values: -7.2E+75 through 7.2E+75
FLOAT	Single-precision floating-point numbers. Storage size: 4 bytes Range of values: -3.40E + 38 through -1.18E - 38, 0 and 1.18E - 38 through 3.40E + 38
INTEGER	Large integers. Storage size: 4 bytes Range of values: -2^31 (-2147483648) through 2^31 - 1 (2147483647)
NVARCHAR(*n*)	Variable-length Unicode data. Valid length values: 1 through 4000 bytes
TIMESTAMP(*f*)	Date and time value that includes the year, month, day, hour, minutes, and seconds. Timestamp format for Tcl scripts: yyyy-mm-dd hh:mm:ss ff Timestamp format for SQL expressions: target database format The *f* value is the number of digits in the fractional part of seconds. Valid fractional seconds precision values: 0 through 9
VARCHAR(*n*)	Variable-length non-Unicode data. Valid length values: 1 through 4000 bytes

5. In the **Expression** list, select the Tcl or SQL expression that you previously defined.
6. Click **OK** to save the column-expression association.

   Tip: To view virtual columns that use a particular expression, in the **Add/Show expressions** dialog box, right-click the expression row and click **List virtual columns**.
7. On the **Map Columns** view, map the virtual source column with the Tcl or SQL expression to the corresponding target column. These columns must have compatible datatypes.

   For more information about recommended datatype mappings for virtual source columns, see the *Informatica Data Replication Datatype Mapping Reference*.

   When you run the InitialSync or the Applier component later, the component evaluates the expression for the virtual column and writes the result to the corresponding target column.

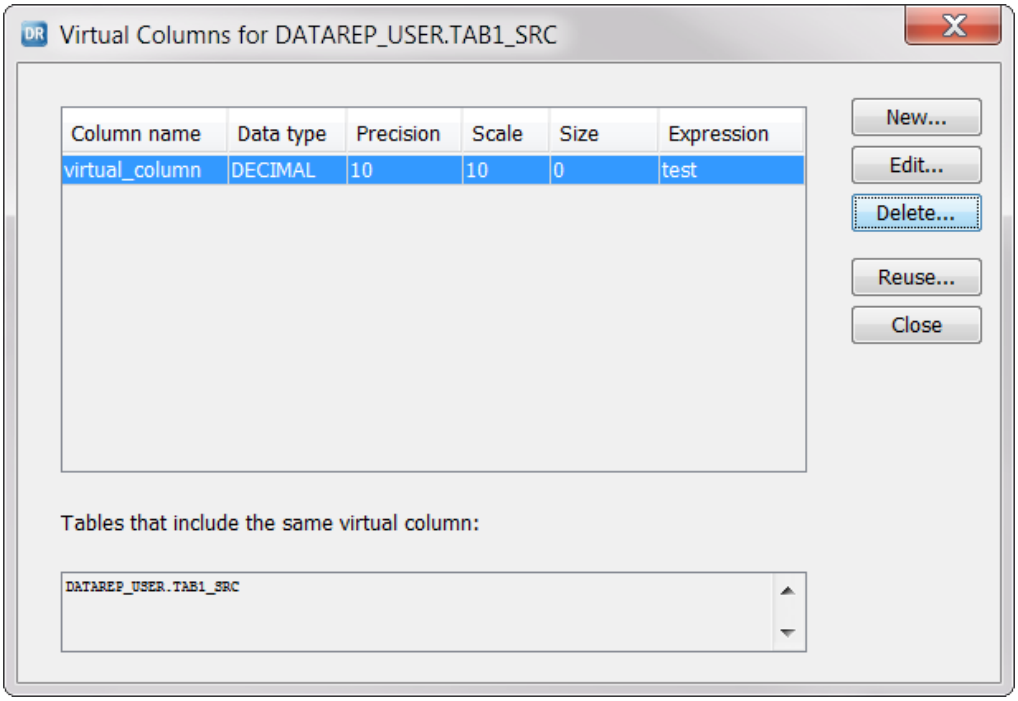
Reusing Virtual Columns in Other Tables

In the Data Replication Console, you can define an expression and virtual column once and reuse them in other tables. This practice helps you save time in implementing an expression across multiple tables.

1. On the **Map Tables** tab, select the mapped source table to which you previously added a virtual column.

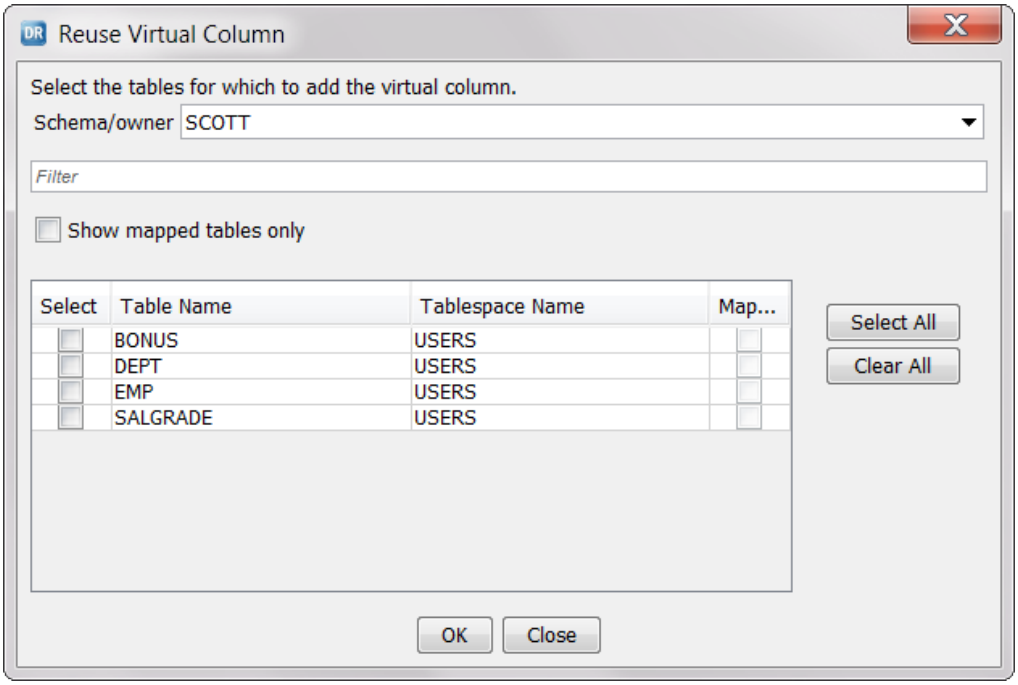
2. Click the **Show Virtual Columns** icon button above the source tables filter field, or right-click the source table row and click **Show Virtual Columns**.

   The **Virtual Columns for <table_name>** dialog box appears.



3. Select the virtual column that you want to reuse and click **Reuse**.

   The **Reuse Virtual Column** dialog box appears.



4. In the **Schema/owner** field, enter the schema or owner name of the source tables in which you want to reuse the virtual column.

5. To filter the list of tables, enter the first few letters of the source table names in the *Filter* field.

   For case-sensitive filtering, enclose the filter value in double quotation marks.

6. To show only the mapped tables, select **Show mapped tables only**.
7. Select the tables in which you want to reuse the virtual column.
   - To select all of the tables, click **Select All**.
   - To select tables individually, select the **Select** check box next to each table name.

   Tip: If you want to deselect all of the tables, click **Clear All**. To deselect one or more tables individually, clear the **Select** check box for each table.

8. Click **OK**.

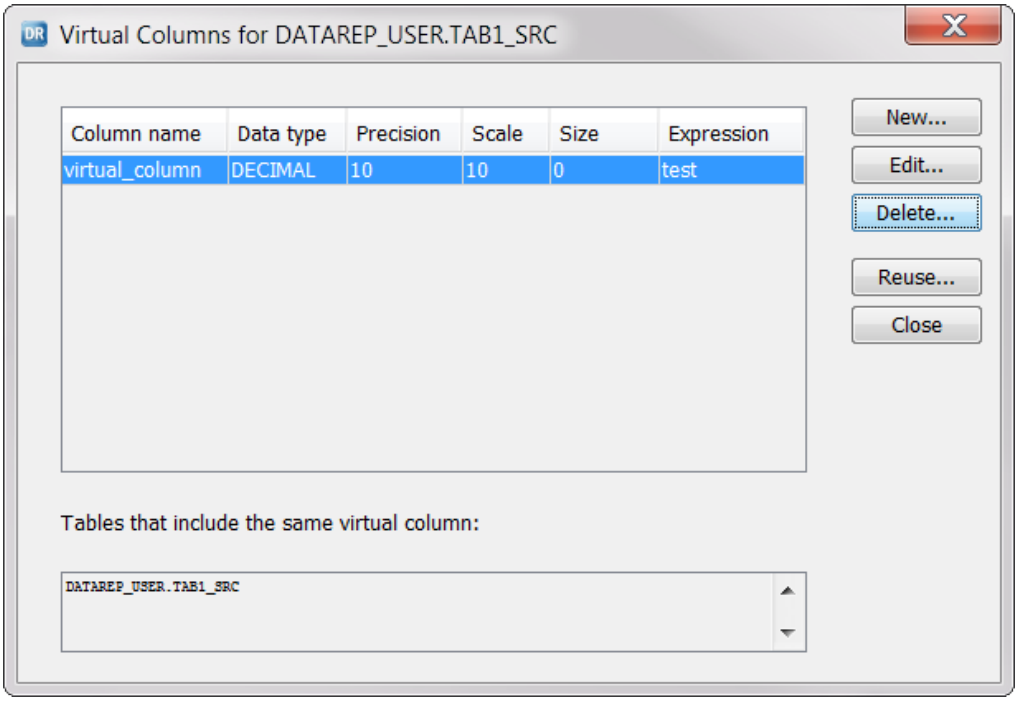
   In the **Virtual Columns for <table_name>** dialog box, the **Tables that include the same virtual column** box shows the tables that you selected in Step 7.

Editing Virtual Columns

In the Data Replication Console, you can edit existing virtual columns.

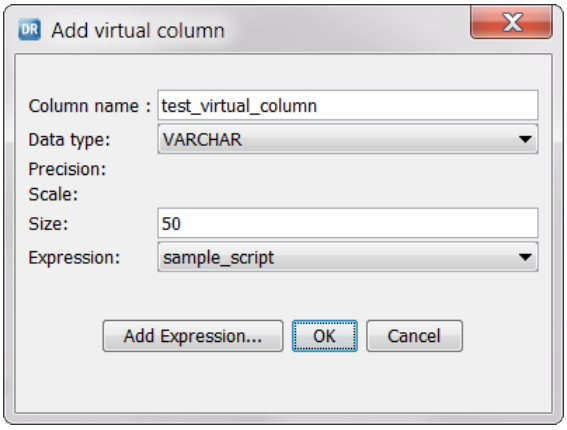
1. On the **Map Tables** tab, select the mapped source table to which you previously added the virtual column that you want to edit.
2. Click the **Show Virtual Columns** icon button above the source tables filter field, or right-click the source table row and click **Show Virtual Columns**.

   The **Virtual Columns for <table_name>** dialog box appears.



3. Select the virtual column that you want to edit and click **Edit**.

The **Add virtual column** dialog box appears.



4. Enter changes in one or more fields.

   The availability of some fields depends on the datatype.

   Important: If you change the datatype of a mapped virtual column, Data Replication validates that the virtual column and target column have compatible datatypes. If the datatypes are not compatible, the Data Replication Console displays an error message that prompts you to map the virtual column to a target column that has a compatible datatype.

5. Click **OK**.

   Data Replication applies the changes. If the replication configuration contains multiple virtual columns that have the same column name, datatype, expression, and properties (precision, scale, and size), Data Replication applies the changes to all of these virtual columns.

Deleting Virtual Columns

In the Data Replication Console, you can delete the virtual columns that you previously added to a replication configuration.

1. On the **Map Tables** tab, select the mapped source table to which you previously added the virtual column that you want to delete.
2. Click the **Show Virtual Columns** icon button above the source tables filter field, or right-click the source table row and click **Show Virtual Columns**.

   The **Virtual Columns for <table_name>** dialog box appears.
3. Select the virtual column that you want to delete and click **Delete**.

   A confirmation prompt appears.
4. Click **Yes**.
5. If other mapped source tables include virtual columns that have the same name, datatype, and properties as the one you deleted, the Data Replication Console asks if you want to delete the identical virtual columns in all of these tables or only the virtual column that you selected in step 3. Click one of the following options:
   - **Yes**. Delete all of the identical virtual columns.
   - **No**. Delete the single virtual column only.

Tip: To delete identical virtual columns from multiple tables selectively, select the virtual column row and click **Reuse**. Clear the **Select** check boxes for the tables from which you want to delete the virtual column and click **OK**.

Specifying the Database Logs from Which to Extract Data

To extract change data from DB2 for Linux, UNIX, and Windows archive logs, Microsoft SQL Server backup logs, MySQL binary logs, or Oracle archived redo logs, you must specify the location of the log files. The log files can be on the local system or a remote system.

Data Replication searches for the last used paths and directories for the SQL Server backup log files, MySQL binary log files, or Oracle archived redo logs and displays these locations on the **Extract Range** tab, if found. For DB2 sources, you must add the directory that contains the DB2 archive log files.

Optionally, you can filter these logs for DB2, Microsoft SQL Server, and Oracle sources. Also, for Microsoft SQL Server and Oracle sources, you can configure extraction processing for database online logs.

1. Click the **Extract Range** tab.
2. To disable change data capture from online logs, clear one of the following options, depending on your source database type:
   - For Microsoft SQL Server sources, clear **Read from online transaction logs**.
   - For Oracle sources, clear **Read from online logs**.

   By default, Data Replication captures change data from online logs for these source types.
3. To add the directory for DB2 archive logs, Microsoft SQL Server backup logs, MySQL binary logs, or Oracle archived redo logs:
   a. Click **Add Directory**.
   b. In the **Add Directory** dialog box, select one of the following options to indicate the location of the log files:
      - **File system**. Use the location of the log files that you specify on the file system of the computer where the source Server Manager runs or on a network mapped share, such as NFS or Samba. This option is the only available option for MySQL sources.
      - **Oracle ASM**. For Oracle sources, use the location of the archived log files in ASM.
      - **Default Archive Logs**. For DB2, Microsoft SQL Server, and Oracle sources, search for the log files in the database-specific location.

      The selected option determines what is displayed in the directory tree.
   c. Specify the directory that Data Replication scans for the log files in one of the following ways:
      - In the **User-defined path** field, type the directory path and press Enter.
      - In the directory tree, select the directory that contains the log files.
   d. For DB2, Oracle, and SQL Server sources, select **Scan subfolders** to recursively search for log files in the subfolders of the specified directory.
   e. In the **Filter files** field or **Base Bin Log File Name** field, enter a mask for the names of the log files to read in the selected directory.

      Note: For MySQL sources, enter a mask for the base names of the binary log files.

      You can use the following wildcard characters:
      - An asterisk (*) to match all characters
      - A question mark (?) to match any single character
      - Square brackets ([ ]) to match only one out of several characters

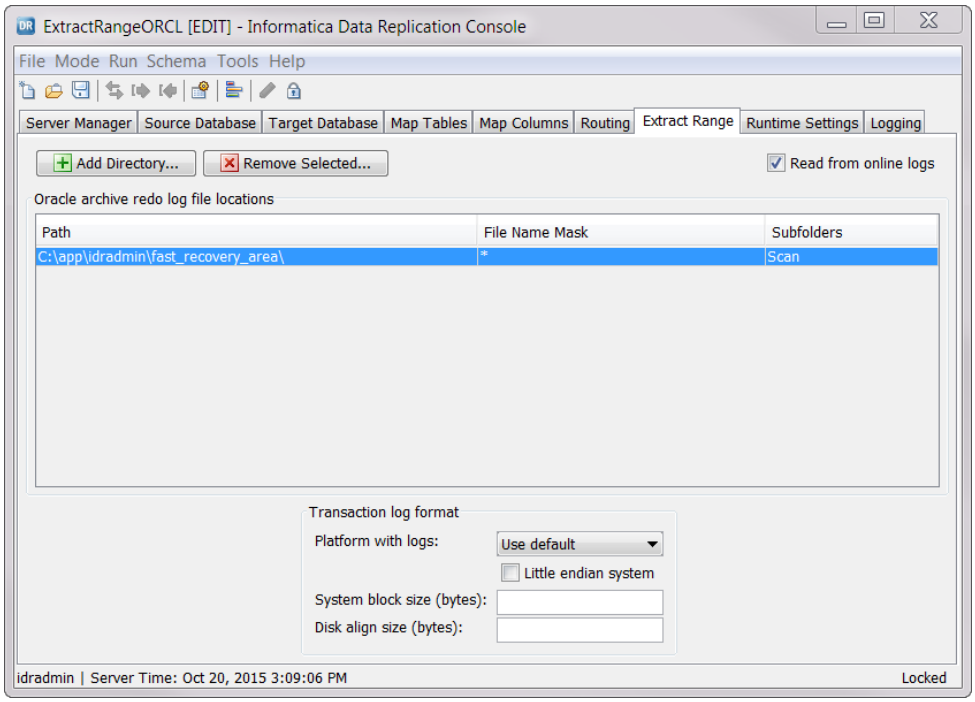
For example, to read all files with the .bak extension, enter *.bak.

Important: The **Filter files** or **Base Bin Log File Name** field cannot be empty. The default is the asterisk (*) wildcard character.

f. Click **OK**.

The directory path, filter mask, and scan subfolders setting are displayed on the **Extract Range** tab.

The following image shows an example of a user-defined path to Oracle archive log files:



Tip: To edit the log path or filter, right-click the path and click **Edit**. To remove the log path and filter, right-click the path and click **Remove**, or select the path and click the **Remove Selected** button.

4. Under **Transaction log format**, specify the DB2 or Oracle platform type and characteristics, which affect log file allocation:
   - In the **Platform with logs** list, select the platform type of the system that contains the source transaction logs. If you do not select a platform type, the **Use default** value remains selected, which causes the Extractor to use the platform on which the source database runs.
   - Select **Little endian system** for a little-endian system. Clear this box for a big-endian system.
   - In the **System block size (bytes)** field, accept the default value that is displayed for the selected platform type or enter another value.
   - In the **Disk align size (bytes)** field, accept the default that is displayed for the selected platform type or enter another value.

   Usually, the default values are acceptable.

Configuring Runtime Settings

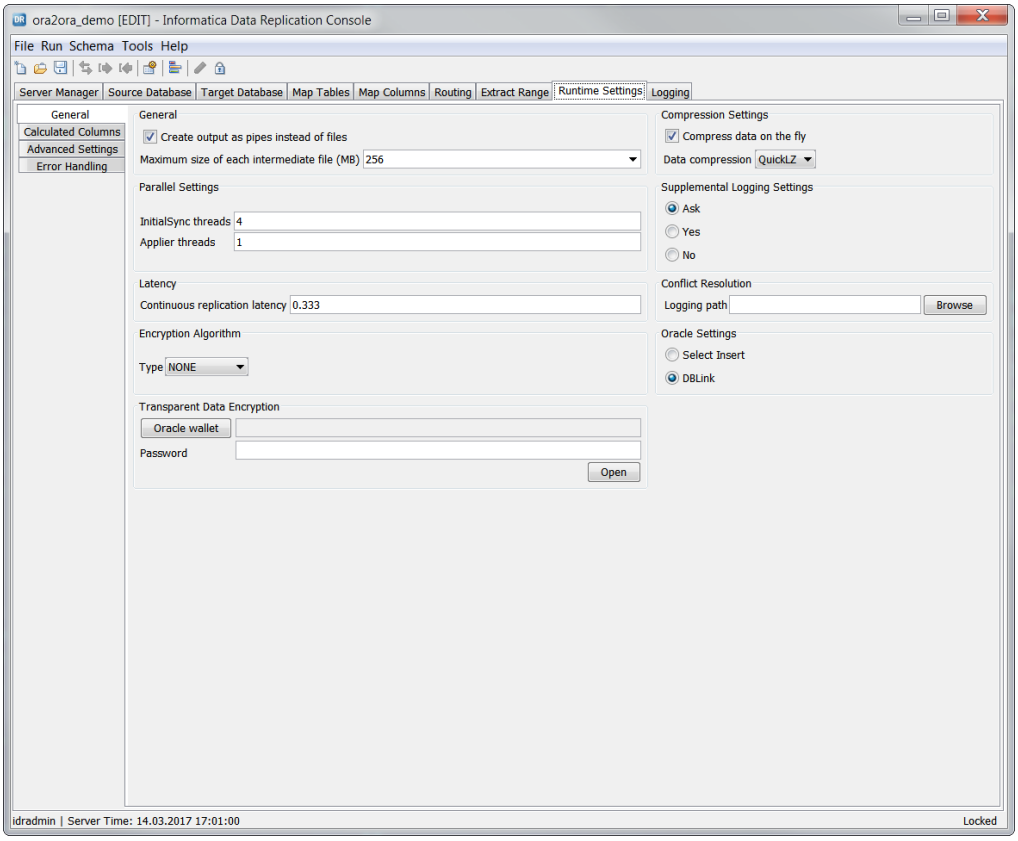
You can specify runtime settings in the configuration file, including advanced settings for high-performance replication jobs.

Specifying General Runtime Settings

You can set general runtime parameters on the **General** view of the **Runtime Settings** tab. These parameters enable important capabilities such as writing output to pipes, compressing data, and multi-threaded processing.

1. Click the **Runtime Settings** tab > **General** view.

   The following image shows the **General** view when the source is Oracle:



2. Enter information in any of the fields, or accept the default values.

The following table describes the fields in the **General** view:

Field	Description	Default
Create output as pipes instead of files	For InitialSync operations and for change data replication in Merge Apply mode to Netezza targets, select this option to have Data Replication write output to pipes instead of files. Data Replication then loads data from the pipes directly into the targets. Clear this option to not write data to pipes.	Selected
Maximum size of each intermediate file (MB)	The maximum amount of data, in megabytes, that you want a single intermediate .DAT file to contain. Options are: **16**, **32**, **64**, **128**, **256**, **512**, and **1024** MB. Data Replication can create a new intermediate file only after the Extractor records a checkpoint. For more information, see "Checkpointing" on page 46. If the Extractor captures a low volume of change data during a cycle, Data Replication might generate an intermediate file that is smaller than the specified maximum size. If the Extractor captures the maximum amount of data but a checkpoint is not triggered, Data Replication generates an intermediate file that is larger than the specified maximum size. For Oracle sources, if you set the extract.use_strict_intermediate_file_size runtime parameter to 1, the Extractor treats the specified maximum size as a hard limit. The Extractor estimates the size of the intermediate file when processing each change record. If the estimated file size exceeds the specified limit, the Extractor forces a checkpoint, writes the data, and closes the intermediate file.	256
InitialSync threads	The number of threads for InitialSync parallel processing.	4
Applier threads	The number of threads for Applier parallel processing.	1

Field	Description	Default
Continuous replication latency	The latency, in seconds, of Extractor change capture processing during continuous data replication. This value determines the duration of an Extractor microcycle. During a microcycle, the Extractor captures changes from database logs, creates an intermediate file, writes changes to the intermediate file, and then sleeps. When the latency period ends, the Extractor begins another microcycle. When setting the latency period, consider how it affects replication performance: - If the latency period is greater than the time it takes the Extractor to read changes from the database logs and write them to an intermediate file, the Extractor sleeps until the end of the latency period. To minimize inactive Extractor sleep periods, set the latency to a low value. - If the latency period is equal to or less than the time it takes the Extractor to read changes from the database logs and write them to an intermediate file, the Extractor continues processing and does not sleep. The Extractor begins a new microcycle as soon as it closes the intermediate file. However, if the latency is too low, the performance of the long-running Extractor task might be degraded because of high disk memory consumption and a high core load. In this case, try increasing the latency value in subsecond increments, such as 0.1. Default is 0.333 second, which is usually acceptable to process all changes in a microcycle. If necessary, you can adjust the latency in sub-second increments. **Note:** After the Extractor creates the intermediate file, the Applier can start reading changes and applying them to the target. Therefore, the Applier might also run during a microcycle. For large transactions with many changes, the Applier waits until the Extractor writes all of the changes to the intermediate file, which might take several microcycles, and then begins apply processing.	0.333
Encryption Algorithm	The encryption algorithm that Data Replication uses to encrypt the intermediate files. Options are: - **NONE**. Do not encrypt the intermediate files. - **TRIPLE_DES**. Use the Triple DES algorithm. - **AES128**. Use the AES algorithm with 128-bit key. - **AES192**. Use the AES algorithm with 192-bit key. - **AES256**. Use the AES algorithm with 256-bit key.	NONE
Oracle wallet	*Oracle sources only.* If you have Oracle sources that use Oracle Transparent Data Encryption (TDE), click this button and browse to the Oracle TDE wallet. The wallet must be in PKCS #12 format.	Not applicable
Password	*Oracle sources only.* If you specified the Oracle TDE wallet, enter the password for the wallet and click **Open**.	No default
Add Certificate	*Microsoft SQL Server sources only.* If you have a SQL Server source database that uses Transparent Data Encryption (TDE), click this button and browse to the .pfx certificate file that you copied to the Data Replication installation directory where the Extractor runs.	Not applicable
Edit Certificate	*Microsoft SQL Server sources only.* If you added a certificate for an encrypted Microsoft SQL Server database and want to change its path or password, click this button.	Not applicable

Field	Description	Default
Delete Certificate	*Microsoft SQL Server sources only.* If you want to delete a certificate that you specified for an encrypted Microsoft SQL Server database, select the certificate and then click this button.	Not applicable
Compress data on the fly	Select this option to compress intermediate files during extraction processing to reduce the file size. Clear this option if CPU resources are restricted and you want to avoid the overhead of compression and decompression.	Selected
Data compression	If you selected **Compress data on the fly**, select the compression method. The only available method is **QuickLZ**, which provides fast compression with a low compression ratio.	QuickLZ
Supplemental Logging Settings	Enables database supplemental logging for source tables when you save the configuration. For Oracle sources, generates a supplemental log group for each source table. For DB2 sources, sets the DATA CAPTURE CHANGES option for each source table. For Microsoft SQL Server Sources, sets Change Data Capture for each source table. To control whether the Console automatically enables supplemental logging, select one of the following options: - **Ask**. Displays a confirmation prompt before the Data Replication Console enables supplemental logging for the source tables. - **Yes**. Enables supplemental logging for the source tables without first displaying a confirmation prompt. - **No**. Does not enable supplemental logging for the source tables. You must enable supplemental logging manually for your source tables.	Ask
Conflict Resolution > Logging path	The path to the log file to which the Applier logs data conflicts that occur when applying change data to a target on which other change data activity is occurring. Click **Browse** to browse to the log file for the system where you run the Applier.	No default

Field	Description	Default
Oracle Settings	*Oracle sources and targets only.* Indicates whether to use database links (dblinks) when running InitialSync for an Oracle source and Oracle target. Options are: - **Select Insert**. Prevents Data Replication from using dblinks. - **DBLink**. Enables Data Replication to use dblinks. This option is a synonym for the initialsync.oracle.use_dblink parameter, which is set on the **Runtime Settings** > **Advanced Settings** view.	DBLink
Disable secondary truncation checkpoint	*Microsoft SQL Server sources only.* Select this option to disable secondary truncation checkpoint management by the Extractor. Select this option in the following cases: - You run the Extractor that reads transaction logs and SQL Server Change Data Capture (CDC) or native transactional replication against the same database. In this case, SQL Server must manage the secondary truncation checkpoint and produce backup logs. - You run two Extractors that read transaction logs against the same database. In this case, only one Extractor can manage the secondary truncation checkpoint. - You run the Extractor that reads transaction logs and a Microsoft SQL Server Backup task for which the **Run the sp_repldone procedure** option is selected. **Important:** After you disable Extractor management of the secondary truncation checkpoint, ensure that the Extractor reads both the transaction and backup logs. Otherwise, incomplete data capture can occur. This option is a synonym for the extract.mssql.together_with_native_replication parameter on the **Runtime Settings** > **Advanced Settings** view.	Not selected

Related Topics:

- "Saving Replication Configurations to the Main Server Manager" on page 267
- "Managing Database Supplemental Logging" on page 342

Enabling Replication from Oracle Tables and Tablespaces That Use Transparent Data Encryption

If you use Oracle TDE to encrypt Oracle source columns and tablespaces, you must open the Oracle wallet that contains the required master keys to enable replication of data from these encrypted objects.

1. On the **Runtime Settings** tab > **General** view, click **Open wallet**.

   The **Open Oracle Wallet** dialog box appears.
2. In the **Server** list, select the Server Manager instance for which you want to open the Oracle wallet.
3. Click the **Browse** button and browse to the Oracle TDE wallet file. Then click **OK**.

   The path to the Oracle wallet appears in the **Path to wallet** field.
4. Click **OK**.
5. In the **Password** field, enter the password for the specified Oracle wallet.
6. Click **Open**.

   The Server Manager instance opens the Oracle wallet, encodes the master keys, and sends the keys to the Data Replication Console.

The Server Manager uses the internal master key to encode master keys from the Oracle wallet. If the Server Manager **Generate Internal Master Key** property is set to 1, the Server Manager Main server creates the internal master key when you open an Oracle wallet for the first time. The path to the internal master key that includes the master key file name is specified in the Server Manager **Internal Master Key Path** property.

Tip: For secure transfer of the internal master key over a network, configure the Server Manager instances to use HTTPS.

7. In the **Select Keys from Oracle Wallet** dialog box, select the key IDs of the master keys that you want to import into the replication configuration and click **OK**.

   The Data Replication Console gets the encrypted TDE table keys from the Oracle source database and then saves them with the encrypted master keys in the replication configuration. If the configuration contains old master keys and table keys, the Data Replication Console deletes the old keys prior to saving the new keys. The list of imported master key IDs appears under **Transparent Data Encryption**.

   By default, after you import master keys from an Oracle wallet, the Extractor ends with an error if it cannot decrypt a record from an encrypted tablespace.

8. If you want the Extractor to skip records from encrypted tablespaces when the wallet that you opened does not include the required master key, set the extract.oracle.skip_encrypted_tablespace_records runtime parameter to 2.

Enabling Replication from Microsoft SQL Server Databases That Use Transparent Data Encryption

To replicate data from a Microsoft SQL Server database that uses Transparent Data Encryption (TDE), you must add the certificate to the Data Replication installation directory and then specify the certificate and its password in the replication configuration. The certificate secures the database encryption key that is required to decrypt data from the database.

1. To get the certificate for the encrypted SQL Server source database and convert it to .pfx format, perform the following substeps:
   a. Export the certificate for the encrypted SQL Server database. Use the following command:

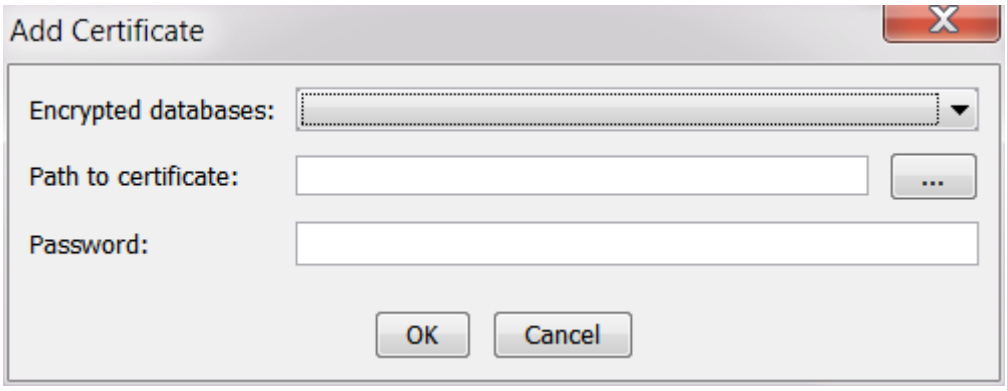
      ```
      BACKUP CERTIFICATE certificate_name TO FILE = 'path_to_.cer_certificate_file'
      WITH PRIVATE KEY
      (
        FILE = 'path_to_.pvk_file',
           ENCRYPTION BY PASSWORD = 'pvk_password'
      )
      GO
      ```

   b. Convert the certificate to .pfx format. Use the following command:

      ```
      pvk2pfx.exe -pvk path_to_.pvk_certificate_file -pi pvk_password -spc
      path_to_.cer_certificate_file -pfx path_to_.pfx_certificate_file -po
      pfx_password -f
      ```

   c. Copy the .pfx certificate to the Data Replication installation directory on the system where the Extractor runs.
2. In the Data Replication Console, on the **Runtime Settings** tab > **General** view, click **Add Certificate** under **Transparent Data Encryption**.

The **Add Certificate** dialog box appears.



3. In the **Encrypted databases** field, select the encrypted Microsoft SQL Server database for which you want to add a certificate.
4. In the **Path to certificate** field, enter the full path to the .pfx certificate file or click the **Browse** button and navigate to the certificate file.
5. In the **Password** field, enter the *pfx_password* that you entered when converting the certificate to .pfx format.

   Important: The Data Replication Console does not verify the password.
6. Click **OK**.

The Data Replication Console adds the information about the certificate to the replication configuration.

Important: Do not delete the .pfx certificate file after you add it to the replication configuration. If you move the .pfx certificate, you must edit the path to the certificate in the configuration.

You can add multiple certificates for a Microsoft SQL Server database. If you create a new certificate for a database, ensure that you add the certificate to the replication configuration. Retain the old certificates in the configuration in case the Extractor needs to process the log files that are encrypted with the old certificate. If the Extractor cannot find a certificate in the configuration that it requires to decrypt a change data record, the Extractor ends with an error.

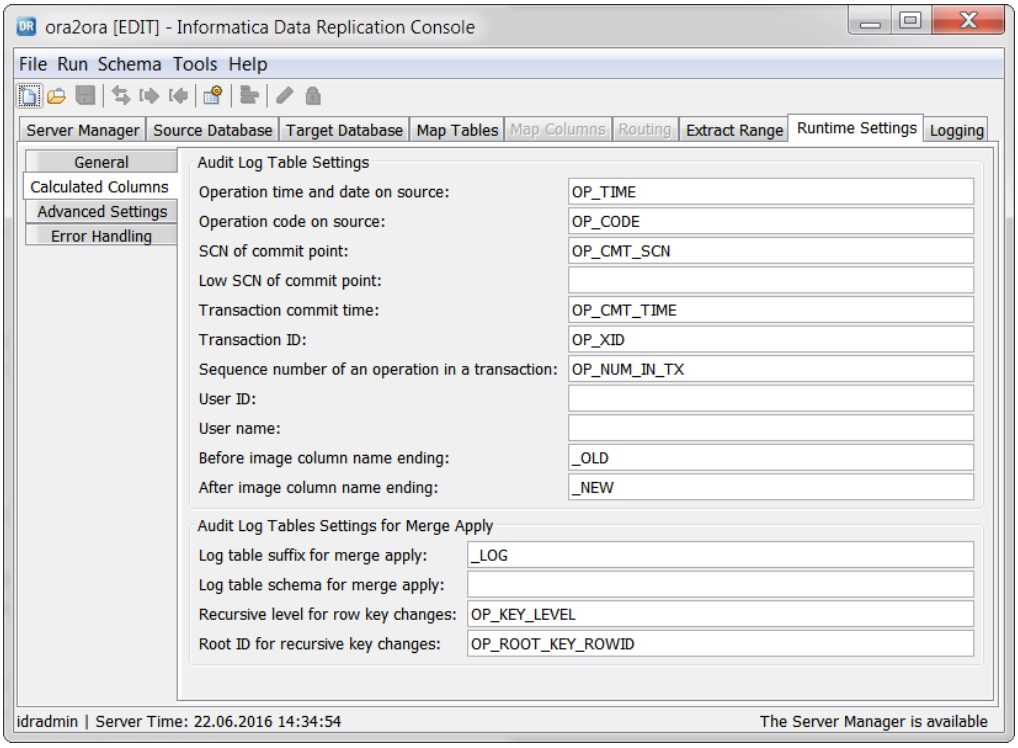
Customizing Names of Calculated Metadata Columns

If you use Audit Apply or Merge Apply mode, you can customize the names of metadata, before-image, and after-image fields that are included in audit log tables, Kafka messages, and the output files that are produced on Cloudera, Flat File, and Hortonworks targets.

The revised settings will be used in the DDL for generating or regenerating the audit log table, Kafka messages, or output files on the target.

1. Click the **Runtime Settings** tab > **Calculated Columns** view.

   The following image shows this view:



2. Enter values in any of the fields, as needed.

   The following table describes the fields on the **Calculated Columns** view:

Field	Description	Default Value
Operation time and date on source	The name of the metadata column that contains the date and time of an operation.	OP_TIME
Operation code on source	The name of the metadata column that contains the operation type code.	OP_CODE
SCN of commit point	The name of the metadata column that contains the SCN of the operation commit.	OP_CMT_SCN
Low SCN of commit point	For Microsoft SQL Server sources, the name of the audit log table metadata column that contains the first part of LSN of the operation commit.	No default

Field	Description	Default Value
Transaction commit time	The name of the metadata column that contains the commit time of the operation.	OP_CMT_TIME
Transaction ID	The name of the metadata column that contains the transaction ID.	OP_XID
Sequence number of an operation in a transaction	The name of the metadata column that contains the sequence number of an operation in a transaction.	OP_NUM_IN_TX
User ID	For DB2, Microsoft SQL Server, and Oracle sources, the name of the audit log table metadata column that contains the user ID of the database user who performed a DML operation, as recorded in the transaction log record for the operation. **Note:** If you leave the field blank, the Data Replication Console does not include this column when generating audit log tables.	No default
User name	For DB2, Microsoft SQL Server, and Oracle sources, the name of the audit log table metadata column that contains the name of the database user who performed a DML operation, as recorded in the transaction log record for the operation. **Note:** If you leave the field blank, the Data Replication Console does not include this column when generating audit log tables.	No default
Before image column name ending	The column name suffix for columns that contain the before-image values for Update and Delete operations. For Merge Apply, the column name suffix for audit log table columns that contain values from after-image primary key columns for Insert operations. **Note:** For Kafka targets, this field is ignored. To change the suffix of the field in Kafka messages that contain the before-image data, use the apply.avro.avro_before_image_suffix runtime parameter.	_OLD

Field	Description	Default Value
After image column name ending	The column name suffix for columns that contain the after-image values for Insert and Update operations. **Notes:** - Either the before- or after-image suffix can be empty but not both. If both the before- and after-image suffixes are empty, the Applier cannot identify the before- and after-image columns in the audit log table. - For Kafka targets, this field is ignored. Kafka messages use the source column names as the names of the columns that contain the after image data.	_NEW
Log table suffix for merge apply	A suffix for audit log table names. - For Merge Apply, the Applier uses this suffix to determine the audit log table that corresponds to each mapped target table. - For Audit Apply, the Data Replication Console uses this suffix to map the source tables to audit log tables based on wildcard expressions or exact name matching. **Notes:** - If the audit log table names with the default suffix of _LOG exceed the maximum table name length that your target database allows, enter a shorter suffix such as _L. - This field is ignored for configurations that do not use Merge Apply.	_LOG
Log table schema for merge apply	When Merge Apply is used, a target schema for audit log tables. **Notes:** - For Netezza targets, leave the default value. You cannot override the schema for audit log tables because a Netezza user can own only one schema. - This field is ignored for configurations that do not use Merge Apply.	No default

Field	Description	Default Value
Recursive level for row key changes	When Merge Apply is used, the name of the audit log table metadata column that contains the first value for a key mutation trace. **Note:** This field is ignored for configurations that do not use Merge Apply.	OP_KEY_LEVEL
Root ID for recursive key changes	When Merge Apply is used, the name of the audit log table metadata column that contains the second value for a key mutation trace. **Note:** This field is ignored for configurations that do not use Merge Apply.	OP_ROOT_KEY_ROWID

3. Save the configuration.

Setting Advanced Parameters for the InitialSync, Extractor, and Applier Tasks

On the **Runtime Settings** tab > **Advanced Settings** view, you can specify runtime parameters for the InitialSync, Extractor, and Applier programs.

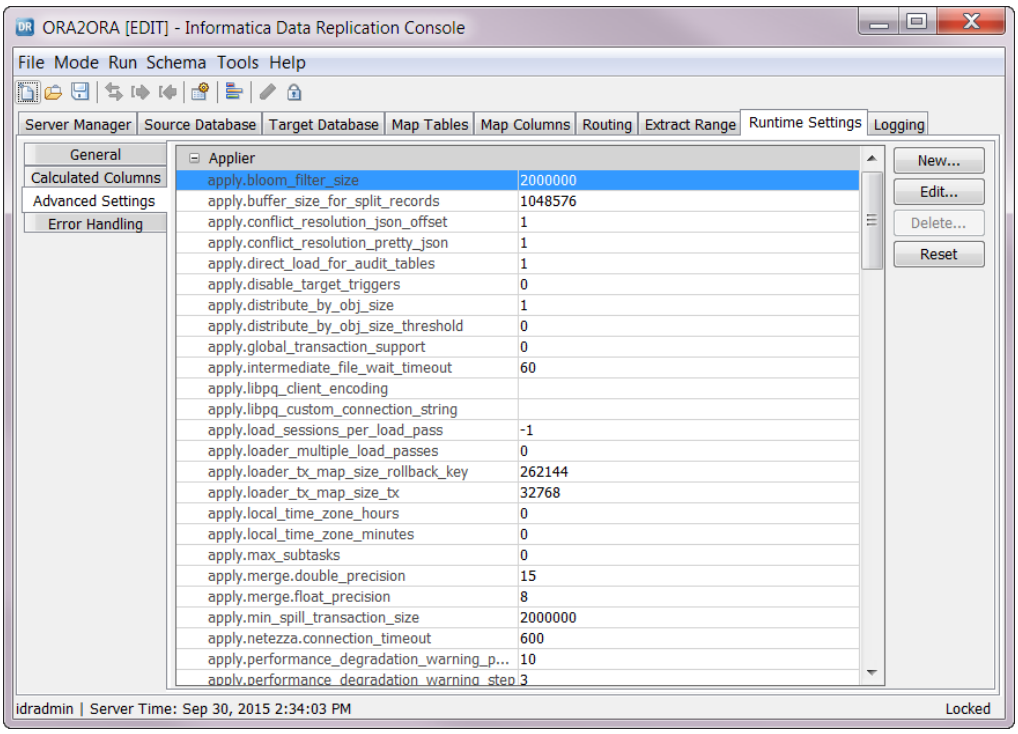
Data Replication supplies default values for most of the parameters listed in the Console and in the AdvancedParameters.xml file in the *DataReplication_installation*/uiconf directory. If you edit the default values or add values for the few parameters that do not have defaults, the updated parameter values are included in the configuration. Also, you might need to add user-defined parameters at the direction of Informatica Global Customer Support.

For descriptions of the parameters that Data Replication supplies, see Appendix C, "Data Replication Runtime Parameters" on page 396.

1. Click the **Runtime Settings** tab > **Advanced Settings** view.

   This view lists the runtime parameters that are applicable to the source and target that are defined for the replication job, including the default parameter values. The parameters are grouped by component or parameter type and listed under each group in alphabetical order.

   The following image shows the **Advanced Settings** view:



2. To edit a listed parameter value, perform the following substeps:

   a. Select the parameter and click **Edit**.

   b. In the **Input** dialog box, enter a new value and click **OK**.

      Data Replication validates the value against the AdvancedParameters.xml file. Data Replication checks that the value is within the minimum and maximum range of values, if applicable, and is consistent with the data type.

3. To reset a listed parameter to its default value, select the parameter and click **Reset**.

   Note: The **Reset** button does not affect any user-defined parameters.

4. If you need to add a user-defined parameter at the direction of Informatica Global Customer Support, perform the following substeps:

   a. Click **New**.

   b. Enter the parameter name that Support supplies and click **OK**.

   c. Enter the parameter value and click **OK**.

      The new parameter appears under the **User Defined** section.

5. To delete a user-defined parameter, select the parameter and click **Delete**.

   Note: You can delete only user-defined parameters.

Configuring Applier Handling of Error Codes

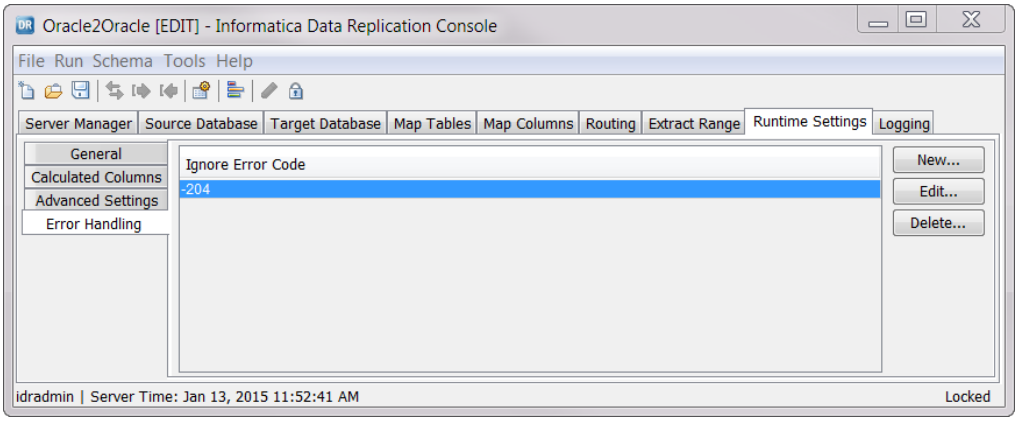
On the **Runtime Settings** tab > **Error Handling** view, you can optionally specify a list of target database and Applier error codes for the Data Replication Applier to ignore during apply processing of DML operations in SQL Apply mode.

Use this feature to force Data Replication to continue apply processing when specific database or Applier errors occur.

Important: When the Applier encounters an error with an error code that is in the **Ignore Error Code** list, the Applier rolls back the current transaction. During the subsequent Applier cycle, the Applier applies each DML operation in the internal buffer individually and skips the DML operations that caused the error. After processing all of the DML operations in the internal buffer, the Applier switches back to the normal mode and applies subsequent DML operations that accumulate in the internal buffer all at one time.

1. Click the **Runtime Settings** tab > **Error Handling** view.

   The following image shows this view:



2. Click **New** to create a rule.

   Data Replication adds a row to the **Ignore Error Code** list box.

3. Enter the database-specific or Applier numeric error code.

   For example, if you enter -204 for a DB2 for Linux, UNIX, and Windows target, the Applier ignores the DB2 -204 return code and continues processing.

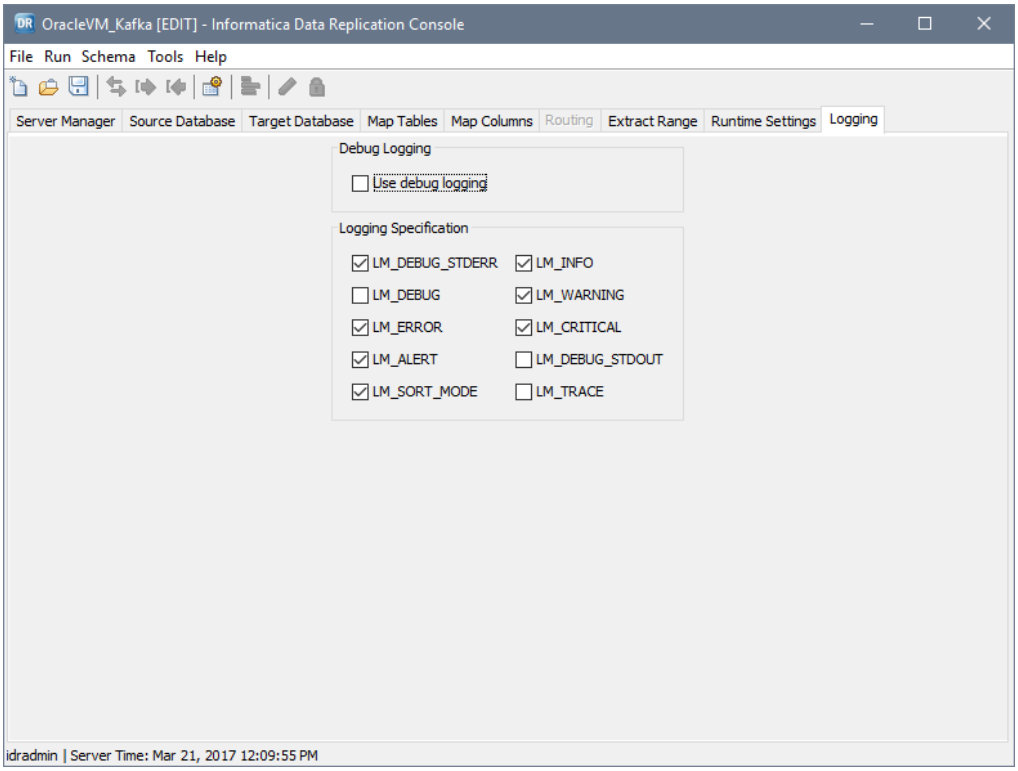
   For Oracle error messages with the prefix ORA, enter only the number that follows the prefix.

Configuring Message Logging

On the **Logging** tab, you can customize the types of messages that Data Replication adds to task execution logs that are located in the *DataReplication_installation*/logs directory.

1. Click the **Logging** tab.

   The following image shows this tab.



2. Leave the **Use debug logging** check box cleared unless Informatica Global Customer Support requests that you select it.

   Note: Selecting this option can add many debug messages that are intended for Customer Support use only to the log file.

3. Select one or more check boxes for the severity levels of messages that you want to log.

   The following table describes each severity level:

Severity Level	Description	Default
LM_DEBUG_STDERR	Debugging information that goes to the standard error output stream.	Selected
LM_DEBUG	Additional debugging information. Select this option only at the request of Informatica Global Customer Support.	Not selected
LM_ERROR	Error messages.	Selected
LM_ALERT	A condition that should be corrected immediately, such as a corrupted database.	Selected

Severity Level	Description	Default
LM_SORT_MODE	For Oracle sources, enables or disables the sorting of change records by the Extractor when it parses redo logs. Clear this option only at the request of the Informatica Global Customer Support for debugging purposes.	Selected
LM_INFO	Conditions that are not error conditions but that might require special handling.	Selected
LM_WARNING	Warning messages.	Selected
LM_CRITICAL	Critical conditions, such as hard device errors.	Selected
LM_DEBUG_STDOUT	Debugging information that goes to the standard output stream. Select this option only at the request of Informatica Global Customer Support.	Not selected
LM_TRACE	Informational messages that are normally used for debugging only. Select this option only at the request of Informatica Global Customer Support.	Not selected

If you select none of the check boxes, the Data Replication Console warns you that no information will be logged.

Saving Replication Configurations to the Main Server Manager

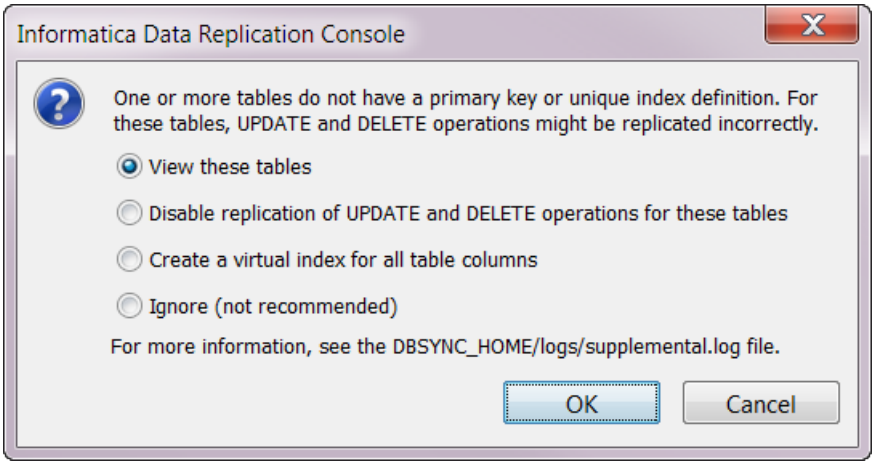
After configuring a data replication in the Data Replication Console, save the configuration to the Server Manager.

1. Click **File** > **Save** on the menu bar, or click the **Save** icon button on the main toolbar.
2. For Oracle sources, if all of the tables in the new configuration have a primary key definition or unique index, the Data Replication Console, by default, displays a prompt that asks if you want to run supplemental logging. Click **Yes** to enable supplemental logging for the source tables in the configuration.

   If you do not have sufficient privileges to manage database supplemental logging in Data Replication, the Console displays a warning message. In this case, save the configuration and follow the instructions in "Managing Oracle Supplemental Log Groups" on page 346 to configure supplemental log groups and then save the generated SQL statements to a script file, which you can give to a database administrator.

   Note: If you click **Yes** and a table includes a unique index that has only nullable columns, the Console enables supplemental logging for the unique index columns so that the Applier can build a WHERE clause to replicate Updates and Deletes. However, the Applier might then apply Update or Delete operations to an incorrect row if a table contains multiple rows with the same value in a unique index column. To prevent this problem, verify that the table does not contain multiple rows in which all index columns contain null values.
3. For DB2 and Oracle sources, if any source table in the new configuration does not have a primary key or unique index definition, the Data Replication Console prompts you to select an option for replicating these tables.

The following image shows this prompt:



Note: The prompt appears only if the **Ask** option under **Supplemental Logging Settings** is selected on the **Runtime Settings** tab > **General** view. The **Ask** option is selected by default.

Options are:

- **View these tables**. Displays only the affected mapped tables on the **Map Tables** tab. A prompt notifies you that enabling supplemental logging was canceled. Click **OK**. You can then assign a primary key or unique index definition to these tables in the source database, or save the configuration again to select a different option from the prompt.
- **Disable replication of UPDATE and DELETE operations for these tables**. Allows you to replicate the configuration without assigning a primary key or unique index for the affected tables. This option minimizes the potential for replication errors, but Data Replication will not apply any Update or Delete operations from the source tables to the target.
- **Create a virtual index for all table columns**. Allows Data Replication to create a virtual index and (for Oracle sources) supplemental log groups for all table columns. Informatica recommends this option. Data Replication uses the following naming pattern to create supplemental log groups for Oracle sources:

  `IDR_<object_ID_of_source_table>_<sequence_number>`

  The incremental sequence number begins with 1 and is used to differentiate between the multiple supplemental log groups that are generated for a single table with more than 33 columns.

  After you save the configuration, you can edit which table columns are included in supplemental log groups on the **Map Tables** tab in the **Manage Oracle Supplemental Log Groups** dialog box.
- **Ignore**. Saves the configuration without changes. Informatica does not recommend selecting this option because Update and Delete operations might be replicated incorrectly for the tables that do not have a primary key or unique index definition.

4. For DB2 sources, the Data Replication Console displays a prompt that asks if you want to manage the DB2 DATA CAPTURE settings for the mapped source tables. Click **Yes** to manage these settings for the tables in the configuration. After you save the configuration, you can edit the DATA CAPTURE settings in the **Manage DB2 DATA CAPTURE Settings** dialog box, which is accessed from the **Map Tables** tab.
5. For targets that use a recovery table, the Data Replication Console displays the **Enter Recovery Table Name and Schema** dialog box. Accept the default schema and table name for the recovery table that will be on the primary target or enter another schema or table name. For some targets, you might need to specify a database or owner name. Then click **Yes**.
   The Data Replication Console sets the apply.recovery_enabled parameter to 1 and the apply.recovery_table parameter to the recovery table name on the **Runtime Settings** tab > **Advanced**

Settings view. When you start the Applier later, it transparently creates the recovery table on the target. For the Applier to create the recovery table, it must run under a user ID that has the privileges required to create a table on the target.

Note: If you click **No**, the recovery table is not created and the apply.recovery_enabled parameter is set to 0. If you want to use a recovery table later, you can create it in one of the following ways:

- Manually create the table by using scripts. For more information, see Appendix F, "DDL Statements for Manually Creating Recovery Tables" on page 455. Then set the apply.recovery_enabled parameter to 1 and set the apply.recovery_table parameter to the schema and table name of the table that you created, before saving the configuration again.
- Edit the configuration to set the apply.recovery_enabled runtime parameter to 1. Then save the configuration again. The **Enter Recovery Table Name and Schema** dialog box appears. Specify the schema and table name of the recovery table and click **Yes**. The Data Replication Console sets the apply.recovery_table parameter to the recovery table name. When you start the Applier later, it will create the recovery table.

The Server Manager saves the configuration on the Main server system.

Related Topics:

- "Managing Database Supplemental Logging" on page 342
- "Specifying General Runtime Settings" on page 253

CHAPTER 11

Materializing Targets with InitialSync

This chapter includes the following topics:

InitialSync Overview

InitialSync is a Data Replication component that uses source tables to materialize targets. Run InitialSync prior to running the Extractor and Applier tasks.

On the system where InitialSync will run, you must install the connectivity software that Data Replication requires for the source database and target database.

Note: InitialSync does not support Apache Kafka, Cloudera, Flat File, and Hortonworks targets. These target types do not require initial synchronization.

Alternatively, if you have a license for Informatica Fast Clone, you can use Fast Clone to perform a high-speed initial load of Oracle data to target systems. Fast Clone works with Oracle sources only. It loads data to most of the same targets that Data Replication supports and also has a data streaming capability for Amazon Redshift, Greenplum, Netezza, Teradata, and Vertica targets. For Greenplum, Netezza, and Teradata targets, DataStreamer is an optional component to load data faster. For Amazon Redshift targets, DataStreamer is a required component that is used transparently and always enabled.

Source Connectivity for Data Unload Operations

InitialSync uses SELECT statements to unload data from the source. Depending on the source type, it uses different connectivity mechanisms to unload data.

The following table describes connectivity mechanisms that InitialSync uses to unload data from different sources:

Source	Software Connectivity
DB2 for Linux, UNIX, and Windows	InitialSync uses the IBM DB2 ODBC driver on Windows or the DataDirect ODBC driver for DB2 on Linux and UNIX.
Microsoft SQL Server	On Windows, InitialSync uses the ODBC driver that the SQL Server Native Client provides. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for Microsoft SQL Server that Data Replication provides.
MySQL	On Windows, InitialSync uses the MySQL ODBC driver. On Linux, InitialSync uses the DataDirect ODBC driver for MySQL that Data Replication provides.
Oracle	InitialSync uses the OCI library. For replication from Oracle sources to Oracle targets, InitialSync can alternatively use DBLinks to unload data.

Target Connectivity for Data Load Operations

The utilities that InitialSync can use to connect to the target to load data depends on the target type and some parameter settings.

For most targets, InitialSync can use either a native load utility or the appropriate ODBC driver to load data. If possible, InitialSync uses the native load utility for better performance. If you want to use the ODBC driver instead of the target load utility, run InitialSync with the DIRECT command line parameter set to n or set the initial.enforce_odbc_load runtime parameter to 1 in the configuration.

With the default parameter settings, InitialSync uses the OCI library to load data to Oracle targets and a native database load utility to load data to other targets if a load utility is available for the target type.

The following table describes the load utilities that InitialSync uses to load data to different target types under alternative DIRECT and initial.enforce_odbc_load parameter settings:

Target	DIRECT=y or initial.enforce_odbc_load=0	DIRECT=n or initial.enforce_odbc_load=1
Amazon Redshift	InitialSync uses the LibPQ library to load data.	On Windows, InitialSync uses the PostgreSQL ODBC driver to load data. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for Greenplum that Data Replication provides.
DB2 for Linux, UNIX, and Windows	Not supported.	On Windows, InitialSync uses the IBM DB2 ODBC driver to load data. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for DB2 for Linux, UNIX, and Windows that Data Replication provides.

Target	DIRECT=y or initial.enforce_odbc_load=0	DIRECT=n or initial.enforce_odbc_load=1
Greenplum	InitialSync uses the LibPQ library to load data.	On Windows, InitialSync uses the Greenplum ODBC driver or the PostgreSQL ODBC driver to load data. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for Greenplum that Data Replication supplies.
Microsoft SQL Server	On Windows, InitialSync uses Bulk Copy Program (BCP) to load data. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for Microsoft SQL Server that Data Replication provides.	On Windows, InitialSync uses the ODBC driver that the SQL Server Native Client provides. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for Microsoft SQL Server that Data Replication provides.
MySQL	On Windows, InitialSync uses the MySQL ODBC driver to load data. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for MySQL that Data Replication provides.	On Windows, InitialSync uses the MySQL ODBC driver to load data. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for MySQL that Data Replication provides.
Netezza	On Windows, InitialSync uses external tables with the Netezza ODBC driver to load data. On Linux and UNIX, InitialSync uses external tables with the DataDirect ODBC driver for Netezza.	Not supported.
Oracle	InitialSync uses the OCI direct path load interface to load data. For replication from Oracle sources to Oracle targets, InitialSync can alternatively use DBLinks to unload data.	InitialSync uses the OCI library to load data. For replication from Oracle sources to Oracle targets, InitialSync can alternatively use DBLinks to unload data.
PostgreSQL	InitialSync uses the LibPQ library to load data.	On Windows, InitialSync uses the Greenplum ODBC driver or PostgreSQL ODBC driver to load data. On Linux and UNIX, InitialSync uses the DataDirect ODBC driver for Greenplum that Data Replication provides.
Teradata	InitialSync uses the TPT Load Operator with the Teradata ODBC driver to load data.	InitialSync uses the Teradata ODBC driver to load data.
Vertica	InitialSync uses the LibPQ library to load data.	Not supported.

Sync Point Value

Data Replication stores a Sync Point value in the configuration for each of the mapped target tables. This value indicates the point up to which InitialSync synchronized the target tables with the source tables.

Initially, a configuration specifies -1 as the Sync Point value for all of the mapped tables. This value indicates that initial synchronization has not been completed for the target tables.

When you run InitialSync, it synchronizes the mapped target tables with the source tables for which the Sync Point value is -1. Data Replication then writes an updated Sync Point value to the configuration for each synchronized table, as follows:

- For Oracle and Microsoft SQL Server sources, InitialSync retrieves the system change number (SCN) or log sequence number (LSN) value from the source database for the initial synchronization transaction and writes this value to the configuration for each synchronized table.
- For MySQL sources, InitialSync retrieves the log coordinates from the source database for the initial synchronization. InitialSync writes this value to the configuration for each synchronized table.
- For DB2 for Linux, UNIX, and Windows sources, InitialSync adds records to the service table, which is named DBSYNC_SYNC_LSN by default, for each synchronized table. When you run the Extractor later, it uses these records to determine the LSN value of the source database up to which the target tables are synchronized with the source tables. The Extractor writes this value to the configuration for each synchronized table.

In the Data Replication Console, you can manually edit the Sync Point value that is stored in a configuration for a mapped table. You might need to edit the Sync Point value for a particular table under the following circumstances:

- You need to resynchronize a target table that was previously synchronized. For example, InitialSync ended abnormally. In this case, you need to truncate the table, set its Sync Point value to -1, and then run InitialSync again.
- You want to skip the InitialSync task and configure the Applier to start change data replication for a table from a specific point in the logs. For example, if you use a third-party utility to synchronize target tables, you can set the Sync Point value to the SCN, log coordinates, or LSN that corresponds to the point from which you want to start change data replication.

Related Topics:

- "Changing the Sync Point for the Applier Task" on page 230
- "DBSYNC_SYNC_LSN Table" on page 64
- "Applier Handling of the Sync Point Value" on page 37

InitialSync Handling of Target Table Constraints

An InitialSync task uses one transaction per target table to load data. When loading data to a target, InitialSync does not preserve the order of transactions on the source.

If some data must be committed before other data, target table key constraints might cause the InitialSync task to fail. For example, if you have two target tables with a primary key (PK)–foreign key (FK) relationship and add a row to the first table that references a nonexistent key in the second table, the InitialSync task fails.

To correctly synchronize target tables that have key constraints, use one of the following strategies:

- Disable constraints on the target tables.
- Run the InitialSync task multiple times for different sets of tables based on the table key constraints. For each run, ensure that the target tables that InitialSync will process do not have PK-FK constraints that depend on tables that have not undergone initial synchronization. Perform the following steps to identify the set of tables to process during an InitialSync run:
  1. From the Data Replication Console, click the **Configure Start Point** button on the **Map Tables** tab.

2. In the **Configure Start Point** dialog box, under **Applier start point**, select the target tables that you want to synchronize.
3. Click the **Edit** icon button. The **Edit Applier SCN**, **Edit Applier Index and Position**, or **Edit Applier LSN** dialog box appears.
4. Click **Resync**.
   The Data Replication Console displays *Resync* in the **Start from SCN**, **Start from Index and Position**, or **Start from LSN** field and sets the Sync Point to -1.
5. For the remaining tables, click the **Get SCN from Source**, **Get Index and Position from Source**, or **Get LSN from Source** icon button to set the Sync Point value to the current SCN, log coordinates, or LSN from the source.

   Tip: To undo all changes that you made for selected tables, click the **Reset** icon button before saving your changes.
6. When you are finished, click **Save**.

When you run InitialSync later, it synchronizes only the tables that have a Sync Point value of -1 and skips synchronizing the remaining tables.

Related Topics:

- "Changing the Sync Point for the Applier Task" on page 230

Considerations for Running InitialSync

Review the following information about running the InitialSync task:

- Typically, you use InitialSync to materialize empty target tables. If a target table is not empty, you can delete all of the data from the table by using the SQL TRUNCATE TABLE statement. If you use the ODBC driver to load data, you can set the initial.check_empty_tables runtime parameter to 1 or 2 to check if the mapped tables are empty.
- If you run InitialSync from the command line, you can use the following command line parameters to selectively synchronize target tables:
  - Set the RESYNC parameter to P to synchronize the tables that are not currently synchronized.
  - Specify the DEST_TABLES or EXCLUDE_DEST_TABLES parameters to filter the tables to synchronize. In this case, you must set the RESYNC parameter to N.
- You can improve InitialSync performance by using multiple threads. You can specify a number of InitialSync threads in the **InitialSync threads** field on the **Runtime Settings** tab > **General** view. A single InitialSync thread loads data to a single target table.
- For a particular configuration, you can run InitialSync and the Extractor at the same time if the set of tables for which you run InitialSync does not intersect with the set of tables for which you run the Extractor.
- Because DB2 for Linux, UNIX, and Windows does not provide a way to retrieve the current LSN from the database, InitialSync inserts a record for each mapped target table in a service table. Data Replication creates the service table with the default name of DBSYNC_SYNC_LSN in the DB2 source database. After you start change data capture, the Extractor uses these records to determine the initial LSN to pass to the Applier. To keep source data consistent, InitialSync locks each source table to prevent write access.
- For Microsoft SQL Server targets, InitialSync uses the snapshot transaction isolation level to provide data consistency. InitialSync records the LSN value of the source data unload transaction.

- For Microsoft SQL Server targets on Windows, InitialSync might run out of memory when it uses the Bulk Copy Program (BCP) to load a large amount of LOB data to the target tables.

  Informatica recommends that you use the ODBC driver for initial materialization of Microsoft SQL Server target tables when you need to load a large amount of LOB data to the tables. To use the ODBC driver, run InitialSync with the DIRECT=n command line parameter.

  If you want to use BCP to materialize tables with a large amount of LOB data, decrease the number of InitialSync threads and the value of the global.lob_truncation_size runtime parameter to avoid running out of memory.
- For Oracle sources, you can use Informatica Fast Clone instead of InitialSync to materialize target tables. Fast Clone provides better performance. For more information, see the Informatica Fast Clone documentation.
- For most Oracle sources, you can use multiple InitialSync subtask threads to unload data. To enable InitialSync multithreaded processing, use the initial.oracle.parallel_subtask_limit and initial.oracle.parallel_sample_percentage runtime parameters.

  If you use the ODBC drivers to connect to targets, InitialSync supports multithreaded load processing for all target types.

  If you use the target native load utilities, InitialSync supports multithreaded load processing for target types other than the following unsupported types:

  - Oracle

  - Teradata

  Also, InitialSync does not support multithreaded processing for the following table types:

  - Oracle source tables that have subpartitions.

  - Source tables that have virtual columns with associated Tcl scripts or SQL expressions.

  If you unload data from these types of tables, ensure that the initial.oracle.parallel_sample_percentage runtime parameter is set to 0.

  For Microsoft SQL Server targets, if you use the BCP utility to load data, verify that the BCP requirements for parallel processing of a table are met. For more information, see the Microsoft SQL Server documentation at the following website:
  http://msdn.microsoft.com/en-us/library/aa196739(v=sql.80).aspx.
- Do not use InitialSync with the following target types, for which initial synchronization is not needed:

  - Apache Kafka

  - Cloudera

  - Hortonworks

  - Flat File
- For Microsoft SQL Server and Oracle sources, before you run InitialSync for a table that was added to an existing replication configuration, ensure that the table does not contain uncommitted data from an open transaction.
  If you run InitialSync for a source table that contains uncommitted data and then start the Extractor, the Extractor might not process the change data from the open transaction after the transaction is committed.

Task Flow: Using InitialSync to Materialize a Target

Perform the following tasks to use InitialSync to materialize target tables:

1. Ensure that you have the correct ODBC drivers installed for your source and target type. For more information about the required ODBC drivers, see the *Informatica Data Replication Installation and Upgrade Guide*.
2. Create a new replication configuration in the Data Replication Console. See "Task Flow: Creating a Replication Configuration" on page 175.

CHAPTER 12

Scheduling and Running Replication Tasks

This chapter includes the following topics:

Methods of Running Replication Tasks

From the Data Replication Console, you can run replication tasks on demand or schedule replication tasks to run at a specified date and time, continuously, or periodically. You can also start or stop a schedule or a task manually.

scheduled replication

Data Replication can run replication tasks within a specified time period and at a specified frequency. You define the time period and frequency when you schedule a replication task by using the Server Manager built-in scheduler in the Data Replication Console. The frequency interval can be in seconds, minutes, weeks, months, or years. The time period is defined by specifying Start and End dates and times.

continuous replication

Data Replication can perform low-latency continuous replication of data for all supported sources, targets, and apply modes. Continuous replication jobs capture and transmit data to intermediate files in configurable short-latency microcycles, usually consisting of milliseconds. In the Data Replication Console, you must schedule continuous replication jobs to run with **Continuous** periodicity and can specify a time period that extends years into the future.

on demand replication

You can start Data Replication tasks from the command line or the Data Replication Console when you need to perform an unscheduled replication on demand.

Types of Replication Tasks

Replication tasks perform data replication component functions and specialized functions such as creating backups of SQL Server transaction logs or running an external program.

You can run replication tasks on demand or schedule replication tasks to run at a specified date and time, continuously, or periodically. You can also start or stop a schedule or a task manually. A task can depend on another task or have no dependencies on other tasks. You can view output logs and statistics from task executions.

You cannot delete standard tasks or edit the task names. When you add new tasks, you can define custom task names that are meaningful in your replication environment.

The following table describes the task types:

Task Type	Description
Applier	Runs the Applier on the Server Manager instance that you specify.
Copy File	Copies files from a source Server Manager instance to a target Server Manager instance. Use this task to transmit backup logs produced by the Microsoft SQL Server Backup tasks between the Server Manager servers. **Note:** The Copy File task checks the copy history of processed files. After you copy a file to a target folder, if you try to copy the same file to the same target folder again, the Copy Task skips the file and does not copy it.
External	Runs an external program on the Server Manager instance that you specify.
Extractor	Runs the Extractor on the Server Manager instance that you specify.
InitialSync	Runs the InitialSync component on the Server Manager instance that you specify.
Microsoft SQL Server Backup	Produces Microsoft SQL Server backup logs in native format on the Server Manager instance that you specify. This task is optional. For more information, see "Adding and Scheduling a Microsoft SQL Server Backup Task" on page 285.
Send File	Transmits the intermediate files between the Extractor and Applier.

Related Topics:

- "Adding a Task" on page 282

Schedule and Task Statuses

On the **Server Manager** tab > **Schedules** view, you can view all schedules and their statuses. You can also view the statuses of individual replication tasks in a schedule.

The following table describes schedule statuses:

Schedule Status	Description
error of creating	The Server Manager Main server failed to create a schedule. To diagnose the problem, see the execution logs for the Server Manager Main server.
failure	One or more tasks in the schedule ended abnormally.
pending	The schedule is awaiting its execution time.
running	One or more tasks in the schedule are running.
stopped	All of the tasks in the schedule either stopped or ended abnormally.
success	All of the tasks in the schedule completed successfully.

To update a schedule status, select the schedule row and click the **Refresh** icon button on the **Schedules** toolbar.

The following table describes task statuses:

Task Status	Description
cancelled	The task was canceled manually, or it was canceled because dependent tasks failed.
closing	The task ended successfully or abnormally, and the Server Manager is compressing the task log files.
error before finish	The task ended successfully, but the Server Manager failed to finalize the process correctly. To diagnose the problem, see the execution logs for the Server Manager server that you configured to run the task.
error of creating	The Server Manager Main server failed to create a task. To diagnose the problem, see the execution logs for the Server Manager Main server.
error of instantiation	The Server Manager Main server failed to instantiate a task. To diagnose the problem, see the execution logs for the Server Manager Main server that you configured to run the task.
error on start up	The Server Manager Main server failed to start a task. To diagnose the problem, see the execution logs for the Server Manager server that you configured to run the task.
failure	The task ended abnormally. To diagnose the problem, see the logs for the task.

Task Status	Description
pending	The task is awaiting its execution time.
running	The task is running.
server unreachable	The Server Manager Main server cannot connect to the Server Manager subserver that runs the task to get the task status.
sleep	The task cannot start because other replication tasks are blocking it or because the Data Replication user is editing the configuration that the task uses in Edit mode. To diagnose a problem, select the task on the **Server Manager** tab > **Schedules** view. In the **Processes** box, the **Blocked** column displays blocking status for replication processes: - **No blocking**. No other replication tasks are blocking the task. - **By config**. The Data Replication user is editing the configuration that the task uses in Edit mode. - **By dependecies**. The task depends on another task that cannot start. - **By task**. Other running replication tasks are blocking the task. To view details, select the replication process and click **Blocking Information**.
stopped	The task was stopped, or it aborted.
success	The task completed successfully.

Conflicting Replication Tasks

Data Replication monitors replication tasks that are running to prevent simultaneous execution of conflicting replication tasks for a particular configuration. Data Replication stores information about active replication tasks in the *DataReplication_installation*/lockDB.db3 SQLite database.

To help you prevent conflicts, the Data Replication Console provides the following restrictions:

- You cannot start a replication task in the Data Replication Console or from the command line if it conflicts with one or more other active tasks that use the same configuration.
- You cannot start a schedule if any running task that uses the same configuration conflicts with the scheduled tasks.

The following table describes, for each type of active replication task that uses a configuration, the types of simultaneous tasks that are conflicting and that are not conflicting:

Replication Task	Conflicting Tasks	Nonconflicting Tasks
InitialSync	- InitialSync - Applier	- Extractor
Extractor	- Extractor	- InitialSync - Applier
Applier	- InitialSync tasks that process the same tables as the active Applier task - Applier	- InitialSync tasks that process tables other than those that the active Applier task is processing - Extractor

You can view a list of active tasks that conflict with a scheduled task on the **Server Manager** tab > **Schedules** view.

Running Replication Executables Manually from the Data Replication Console

In the Data Replication Console, you can start the Extractor, Applier, and InitialSync executables individually. The Data Replication Console determines which executable to use for a replication task, depending on the operating system and source and target types that you use. For more information about executables that Data Replication runs for different sources and targets, see "Executables Called from the Data Replication Console or Scripts" on page 390.

In the Data Replication Console, you can run replication executables for the configurations for which are you designated as the owner. The idradmin user can run replication executables for all of the configurations.

To run an executable, use the following methods:

- To run InitialSync, click the **Synchronize** toolbar button or click **Run** > **Synchronize** on the menu bar.
- To run the Extractor, click the **Extract Data** toolbar button or click **Run** > **Extract Data** on the menu bar.
- To run the Applier, click the **Apply to Target** toolbar button or click **Run** > **Apply to Target** on the menu bar.

By default, Data Replication uses standard tasks to run replication executables. For a particular configuration, you cannot override a default task with a custom task that you previously defined on the **Server Manager** tab > **Tasks** view.

Related Topics:

- "Setting Default Tasks for a Configuration" on page 291

Scheduling Replication Tasks

The Data Replication Console provides a built-in scheduler for scheduling replication tasks.

You can schedule the following types of tasks:

- Applier
- Copy File
- External
- Extractor
- InitialSync
- Microsoft SQL Server Backup
- Send File

The scheduler supports a distributed deployment in which the Extractor is installed on one machine and the Applier is installed on another machine. In this case, the intermediate files that the Extractor produces must be sent to the machine where the Applier can retrieve them for apply processing.

You can create a schedule to run a sequence of tasks. For a distributed replication deployment, you typically schedule the following sequence of tasks:

1. An Extractor task
2. A Send File task, as a dependent task of the Extractor task
3. An Applier task, as a dependent task of the Send File task

Note: If you use the Schedule wizard to create a schedule, the task dependencies are defined automatically. The Schedule wizard also creates all of the required tasks.

The idradmin user can schedule and run any replication schedule and task. Regular replication users have restricted privileges to run replication schedules and tasks:

- Regular users can schedule replication tasks for the configuration for which they are designated as the owner.
- Regular users can start and stop a schedule for which they are designated as the owner.

Note: You can also manually run replication tasks, outside of a schedule. Regular users can manually start or stop a replication task for the configurations for which they are designated as the owners.

Related Topics:

- "Configuring Continuous Replication" on page 296

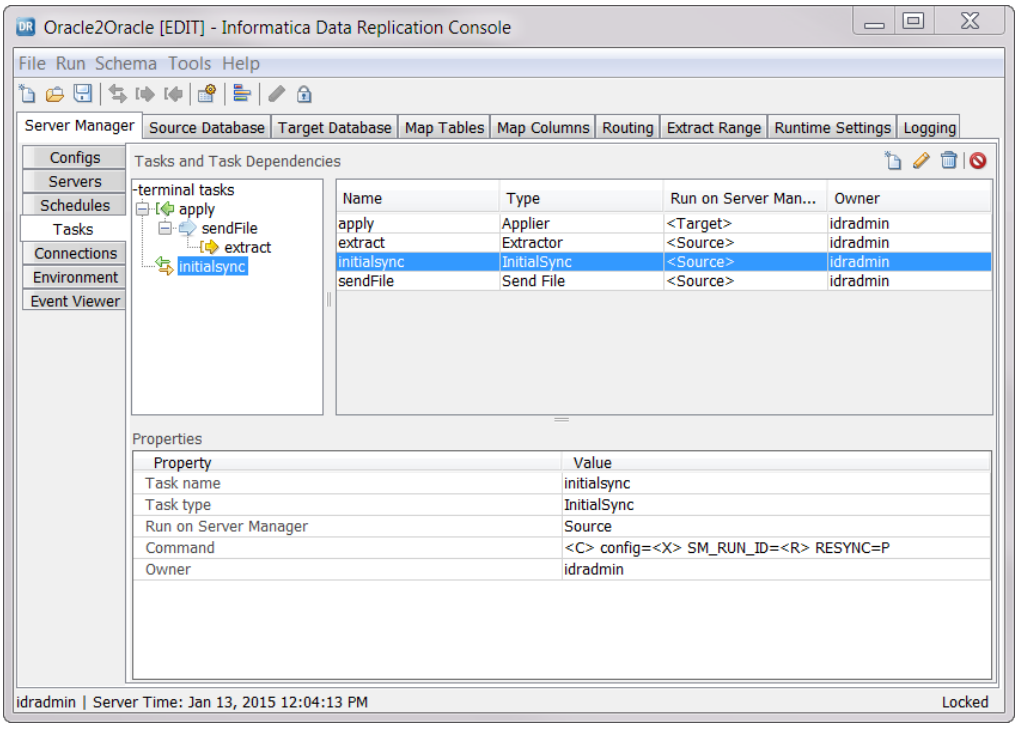
Adding a Task

Before you can schedule a task, you must add it.

Task types include Applier, Copy File, Extractor, External, InitialSync, Microsoft SQL Server Backup, and Send File. A task can depend on another task or have no dependencies on other tasks.

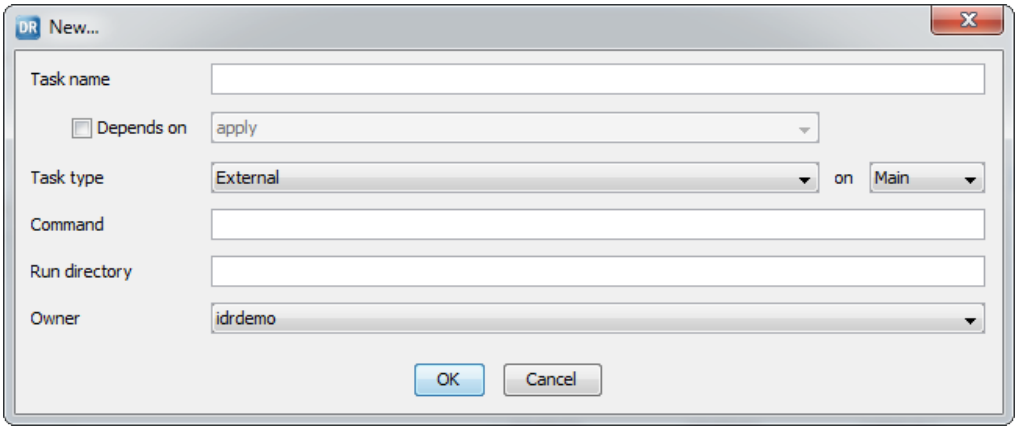
1. Click the **Server Manager** tab > **Tasks** view.

   The following image shows the **Tasks** view:



2. Click the **New** icon button on the Tasks and Task Dependencies toolbar.

The **New** dialog box appears.



3. In the **Task name** field, enter a name for the task.

   Task names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates task names that are longer than 100 characters.

4. If you are defining a task that depends on another task, select **Depends on**. Then select the task on which the new task depends in the adjacent list.

   If you are defining an independent task, ensure that the **Depends on** check box is cleared.

   Important: If dependent tasks require a replication configuration, ensure that the root task on which they depend also requires a configuration. The root task is specified in the **Tasks and Task Dependencies** box on the **Server Manager** tab > **Tasks** view. The Data Replication Console cannot run dependent tasks that require a configuration if the referenced root task does not require a configuration.

5. In the **Task type** list, select the task type. Depending on the selected task type, you might need to enter additional information in the associated fields.

The following table describes the task types and indicates whether you need to enter values in other associated fields:

Task Type	Description and Associated Fields
Applier	Runs the Applier on the server that you select from the list. Data Replication uses the configuration file to build the Applier command and displays the command in the **Command** field.
Copy File	Copies files from a source server to a target server. In the **Source folder** field, enter the path to the folder that contains the file that you want to copy. To browse to the folder, click the **Browse** button. In the **Target folder** field, enter the path to the folder to which to copy the file. To browse to the folder, click the **Browse** button. In the **File Mask** field, optionally enter a file name mask that includes wildcards to copy all of the files that match the mask. For example, if you enter the mask `*.txt`, Data Replication copies all text files from the selected source folder to the selected target folder. Select the **Move** check box to move the files instead of copying them. Use this task to transmit backup logs produced by the Microsoft SQL Server Backup tasks between the Server Manager servers. **Note:** The Copy File task checks the copy history of processed files. After you copy a file to a target folder, if you try to copy the same file to the same target folder again, the Copy Task skips the file and does not copy it.
External	Runs an external program on the server that you select from the list. In the **Command** field, specify the command to run the program. In the **Run directory** field, specify the directory from which to run the command.
Extractor	Runs the Extractor on the server that you select from the list. Data Replication uses the configuration file to build the Extractor command and displays the command in the **Command** field.
InitialSync	Runs the InitialSync component on the server that you select from the list. Data Replication uses the configuration file to build the InitialSync command and displays the command in the **Command** field.
Microsoft SQL Server Backup	Produces Microsoft SQL Server backup logs in native format on the specified server. This task is optional. Enter the Microsoft SQL Server instance name in the **SQL Server instance** field, and enter the database name in the **SQL Server database** field. In the **Target folder** field, specify the name of the directory to which to write the backup files. After adding the Microsoft SQL Server Backup task and including it in a configuration, add this directory on the **Extract Range** tab if the directory is not already listed as a backup log file location for the configuration. If you run native Microsoft SQL Server replication and Data Replication jobs concurrently, clear the **Run the sp_repldone procedure** check box to prevent incomplete data capture by SQL Server. For more information, see "Adding and Scheduling a Microsoft SQL Server Backup Task" on page 285.
Send File	Transmits the intermediate files between the Extractor and Applier.

6. In the **Owner** list, select a task owner.
7. Click **OK**.

8. On the **Server Manager** tab > **Schedules** view, schedule the backup task. See "Creating a Schedule from the Server Manager Tab" on page 286.

   Alternatively, start the task manually. See "Starting a Task Manually from the Tasks View" on page 293.

Adding and Scheduling a Microsoft SQL Server Backup Task

You can optionally add and schedule a Microsoft SQL Server Backup task to back up Microsoft SQL Server logs in native SQL Server format. Data Replication can then capture data from these backup logs.

Important: The Server Manager instance that starts the Microsoft SQL Server Backup task must run on the source database server.

For backup log creation, Data Replication uses the user account that is specified for enabling Change Data Capture. Data Replication uses the native Microsoft SQL Server procedure to produce the backup logs.

Before you begin, in Microsoft SQL Server Management Studio, verify that the SQL Server database is using the full recovery model. You can set the database Recovery Model option to Full in the SQL Server Management Studio or run the following SQL statement:

```
ALTER DATABASE [database_name] SET RECOVERY FULL
```

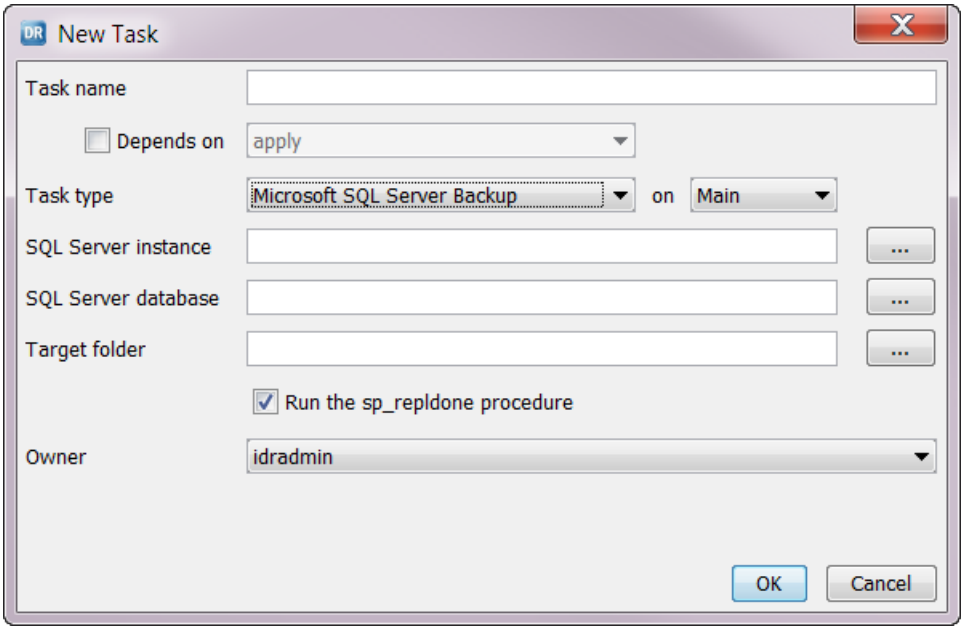
Note: If you back up transaction logs from the SQL Server Management Studio or with another tool that can create uncompressed backups in native SQL Server format, you can configure Data Replication to capture changes from those logs. Use of the Data Replication Microsoft SQL Server Backup task is optional.

If you run both Data Replication and native SQL Server replication concurrently, use the native SQL Server backups.

1. On the **Server Manager** tab > **Tasks** view, click the **New** icon button on the Tasks and Task Dependencies toolbar.

   The **New Task** dialog box appears.
2. In the **Task name** field, enter a name for the Microsoft SQL Server Backup task.
3. In the **Task type** list, select **Microsoft SQL Server Backup**.

   The following image shows the **New Task** dialog box for the Microsoft SQL Server Backup task:



4. In the **SQL Server instance** field, enter or browse to the SQL Server instance.
5. In the **SQL Server database** field, enter or browse to the SQL Server database.
6. In the **Target folder** field, browse to the directory to which to write the backup logs. The Extractor will read the logs at this location.

   Note: After adding the Microsoft SQL Server Backup task and including it in a configuration, add this directory on the **Extract Range** tab if the directory is not already listed as a backup log file location for the configuration.
7. If you run native Microsoft SQL server replication and Data Replication replication jobs concurrently, clear the **Run the sp_repldone procedure** check box to prevent incomplete data capture by SQL Server.

   Select this check box to run the SQL Server sp_repldone stored procedure prior to backing up the online log. This procedure marks all transactions in the online log as replicated to the distribution database. Data Replication then backs up the replicated data to prevent unlimited growth of the online log. However, native SQL Server replication might then get incomplete data from the online log. Because native SQL Server replication does not read data from backup logs, it might lose some replicated data.
8. In the **Owner** list, select an owner for the backup task.
9. Click **OK** to create the Microsoft SQL Server Backup task.
10. On the **Server Manager** tab > **Schedules** view, schedule the backup task. See "Creating a Schedule from the Server Manager Tab" on page 286.

    Alternatively, start the task manually. See "Starting a Task Manually from the Tasks View" on page 293.

    Tip: To avoid data conflicts and loss, run one Microsoft SQL Server Backup task at a time.

Related Topics:

- "Specifying the Database Logs from Which to Extract Data" on page 251

Creating a Schedule from the Server Manager Tab

You can create a schedule for running the tasks that are defined on the **Server Manager** tab > **Tasks** view.

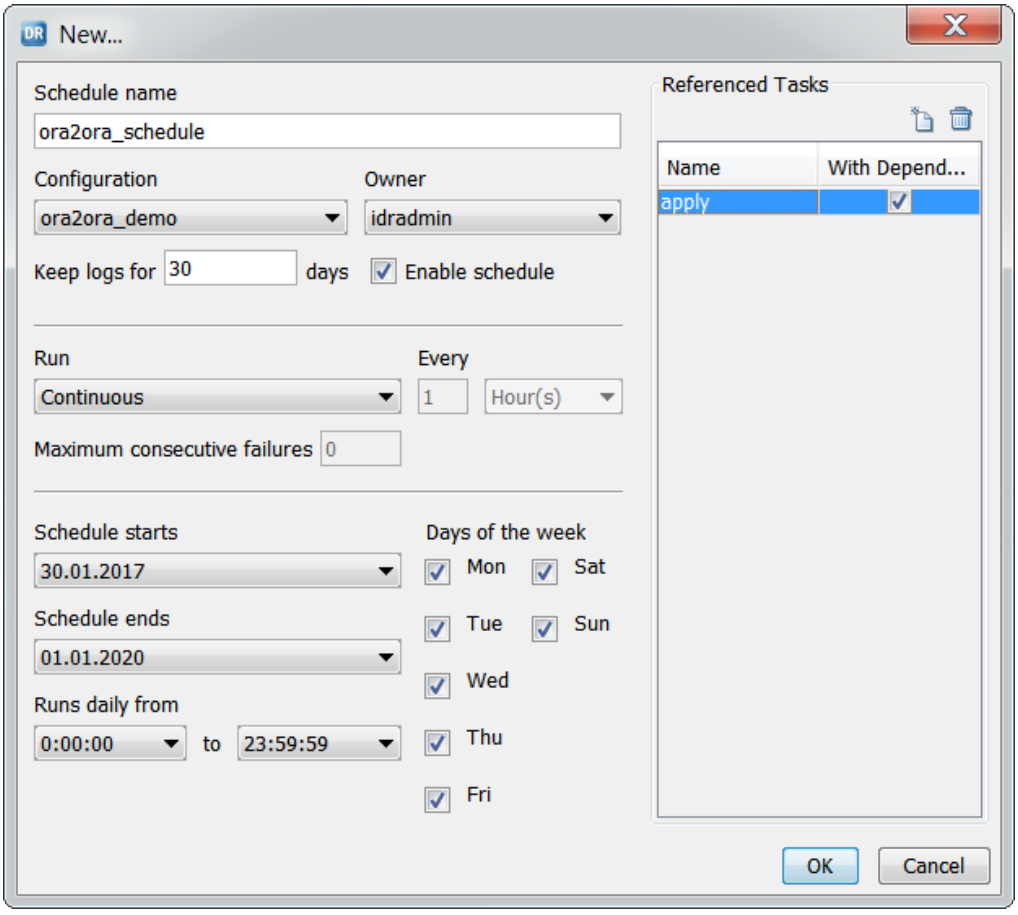
Within a schedule, you can include a particular task only once. Data Replication executes the task and ignores any additional occurrences of it in the schedule.

When you replicate data to multiple targets, the Applier, Send File, and InitialSync tasks contain multiple target processes, one for each target. Otherwise, a task contains a single target process.

You can create a schedule to run replication tasks with a particular configuration or to run tasks that do not require a configuration. You can create a schedule only for the configurations for which you are designated as the owner.

1. On the **Server Manager** tab > **Schedules** view, click the **New** icon button on the Schedules toolbar.

   The **New** dialog box appears.



2. In the **Schedule name** field, enter a unique and descriptive name for the schedule.

   Schedule names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates schedule names that are longer than 100 characters.

3. In the **Configuration** list, select the configuration for the tasks that you are scheduling.

   Data Replication uses this information to generate a command to execute the tasks.

4. In the **Owner** list, select a schedule owner.

5. In the **Keep logs for** field, specify the number of days to retain the output logs from the scheduled Extractor, Applier, and other tasks.

6. Select **Enable schedule** to activate the schedule.

7. In the **Run** list, select the periodicity with which you want to run the scheduled tasks. Options are:

   - **Periodically**. Run the tasks periodically based on other settings such as the date and time at which a task starts and ends, the task run frequency, and days of the week.
   - **On demand**. Run the tasks one time.

- **Continuous**. Perform continuous data replication. The tasks run continuously during the period that you specify in the **Schedule starts** and **Schedule ends** fields.

Note: Do not schedule an InitialSync task to run periodically or continuously.

8. For a schedule that runs periodically, perform the following substeps:
   a. Specify the frequency of execution under **Every**.

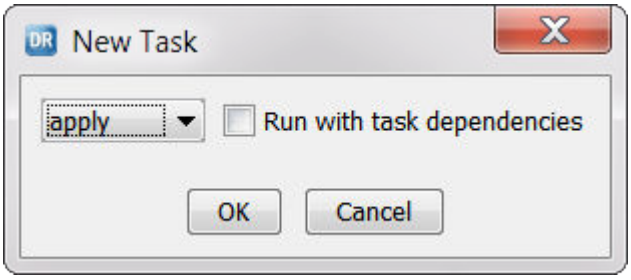
      You can enter a number of seconds, minutes, hours, days, or weeks.

   b. In the **Maximum consecutive failures** field, specify the maximum number of consecutive times that a schedule can end with an error before the Server Manager disables the schedule.

      To allow an unlimited number of consecutive schedule failures, enter 0.
9. In the **Schedule starts** and **Schedule ends** fields, define the period during which the schedule runs. Click in each field to display a control for setting the date.

   For continuous replication, you can define a very long period, up to December 31, 2037.
10. Under **Days of the week**, select one or more days of the week when the tasks run.
11. Under **Runs daily from**, define the start and end times for the schedule run.
12. Under the **Referenced Tasks** box, click the **New** icon button to add a task reference to the schedule.

    The **New Task** dialog box appears.



13. Select a previously defined task from the list.

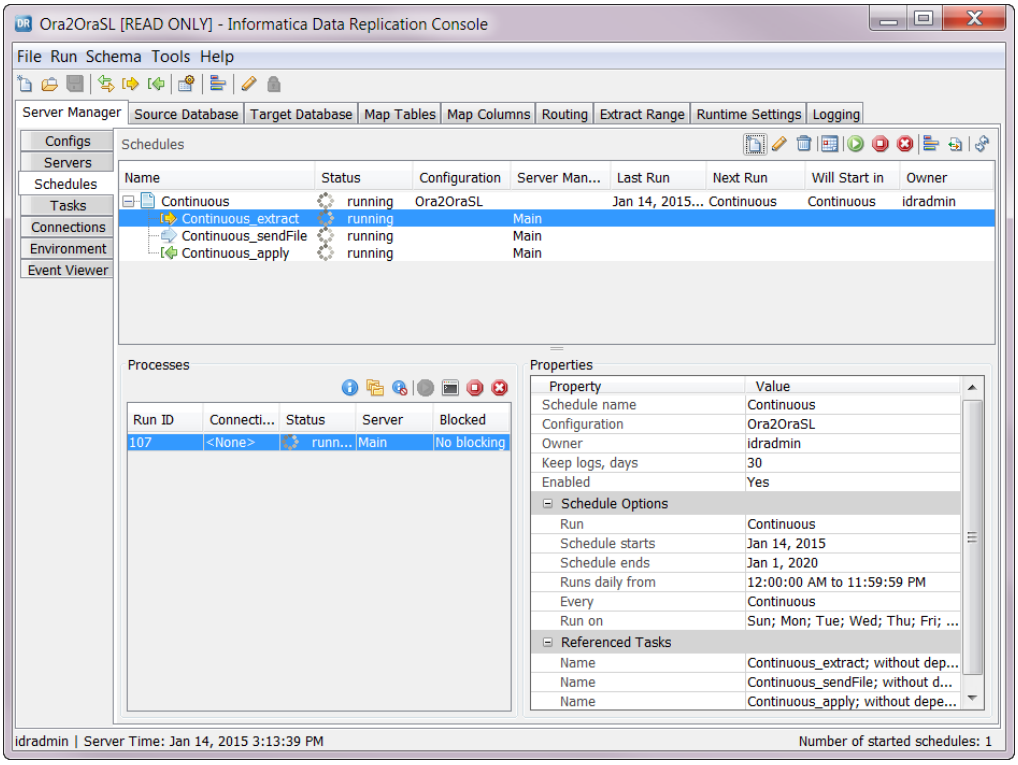
    Note: You can add tasks for which another user is the owner.
14. Select **Run with task dependencies** if the task depends on another task.
15. Click **OK**.

    The task appears in the **Referenced Tasks** box.

    Tip: You can click the **Delete** icon button to remove a task.
16. Repeat steps 12 through 15 to add additional tasks to the schedule.
17. Click **OK**.

    The schedule and its tasks appear in the **Schedules** view.

The following image shows an example view:



Note: If you observe Daylight Savings Time, restart the Server Manager after each time change to ensure that the Data Replication Console displays the accurate time.

The schedule will run based on the scheduling options you set.

18. If you want to start the schedule and run its tasks immediately, click the **Start** icon button on the Schedules toolbar, or right-click the schedule row and click **Start**.

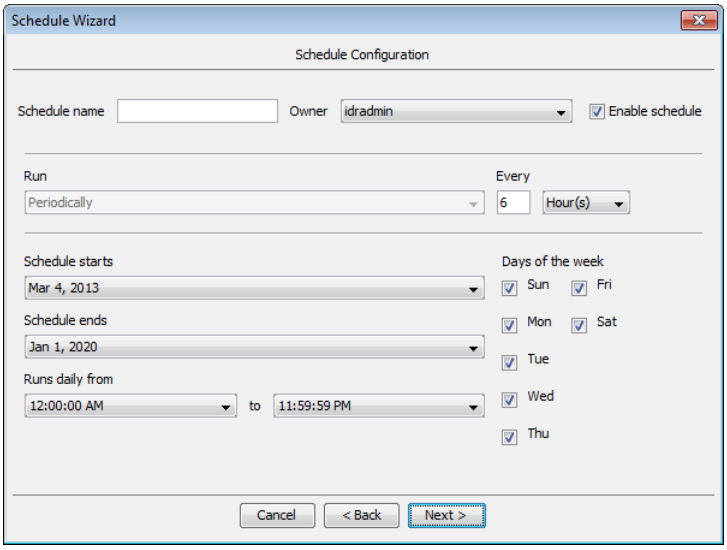
    The **Processes** box shows the status of processes for the Extractor, Send File, Applier, and other tasks.

Creating a Schedule with the Schedule Wizard

For convenience, you can use the **Schedule** wizard to create a schedule.

1. On the **Server Manager** tab > **Schedules** view, click the **Wizard** icon button on the Schedules toolbar to start the **Schedule** wizard.
2. On the **Schedule Type** page, select one of the following execution schemes:
   - **Sequential**. This execution scheme defines one schedule that runs the Extractor, transmits the intermediate file, and runs the Applier sequentially.
   - **Parallel**. This execution scheme defines two schedules that run independently. The first schedule runs the Extractor and transmits the intermediate file. The second schedule transmits the intermediate file and runs the Applier.
   - **Continuous**. This execution scheme defines one schedule that runs the Extractor, transmits the intermediate file, and runs the Applier for continuous replication.
3. Click **Next**.

If you selected the **Sequential** or **Continuous** execution scheme, the **Schedule Configuration** page appears.



If you selected the **Parallel** execution scheme, the **Extractor Schedule Configuration** page appears and the following schedule options in steps 4 to 10 apply to the Extractor and Send File tasks.

4. In the **Schedule name** field, enter a unique and descriptive name for the schedule.

   Schedule names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates schedule names that are longer than 100 characters.

5. In the **Owner** list, select a schedule owner.
6. Select **Enable schedule** check box to activate the schedule.
7. For the **Sequential** and **Parallel** execution schemes only, perform the following substeps:

   a. Specify the frequency of execution under **Every**.

      You can enter a number of seconds, minutes, hours, days, or weeks.

   b. In the **Maximum consecutive failures** field, specify the maximum number of consecutive times that a schedule can end with an error before the Server Manager disables the schedule.

      To allow an unlimited number of consecutive schedule failures, enter 0.

8. In the **Schedule starts** and **Schedule ends** fields, define the period during which the schedule runs. Click in each field to display a control for setting the date.

   For continuous replication, you can define a very long period, up to December 31, 2037.

9. Under **Runs daily from**, define the start time and end time for the schedule run.
10. Under **Days of the week**, select one or more days of the week on which the tasks run.
11. For the **Parallel** execution scheme only, click **Next** and enter the schedule options for the Send File and Applier tasks.
12. Click **Next**.

    The **Replication Configuration** page appears.

13. Select a configuration and click **Next**.

    The wizard generates the schedule.

14. Click **Finish**.

    The **Schedules** view displays the schedule in the list of schedules and scheduled tasks.

Note: If you observe Daylight Savings Time, restart the Server Manager after each time change to ensure that the Data Replication Console displays the accurate time.

The schedule will run based on the scheduling options you set.

15. If you want to start the schedule and run its tasks immediately, click the **Start** icon button on the Schedules toolbar, or right-click the schedule row and click **Start**.

    The **Processes** box lists target processes for the task.

Editing Tasks

You can edit the tasks that are listed in the **Server Manager** tab > **Tasks** view except for the standard tasks that the Data Replication Console provides. The specific tasks that you can edit depend on whether you are the idradmin user.

No user can edit the standard tasks that have the following names: apply, extract, initialsync, and sendFile. However, user-defined tasks are editable, including tasks that have the task type of Applier, Extractor, InitialSync, and SendFile.

The idradmin user can edit all of the tasks except the standard tasks. Regular users can edit only the tasks for which they are designated the owner and that are editable.

Before you edit a task, ensure that the task is not running.

1. Click the **Server Manager** tab > **Tasks** view.
2. In the **Task and Task Dependencies** box, select the task that you want to edit.
3. Click the **Edit** icon button on the Task and Task Dependencies toolbar.

   The **Editing** dialog box appears.
4. Enter the changes.

   You can edit the task name, server, and any other field that is displayed based on the task type.
5. Click **Save**.

   You can also edit task dependencies by dragging tasks in the hierarchical tree that is displayed in the left list box.

Setting Default Tasks for a Configuration

By default, Data Replication assigns the standard Apply, Extract, and InitialSync tasks to configurations. You cannot edit the standard tasks that Data Replication provides. However, if you previously defined a custom task with the type of Applier, Extractor, or InitialSync, you can select that task as the default task for a configuration.

Note: If you are a regular replication user, you can set default tasks only for configurations that you own. If you are the idradmin user, you can set default tasks for any configuration.

1. On the **Server Manager** tab > **Configs** view, select a configuration and click the **Set Default Tasks** icon button on the Replication Configurations toolbar, or right-click a configuration and click **Set Default Tasks**.

   The **Set Default Tasks** dialog box displays the names of the default Extractor, Applier, and InitialSync tasks for the configuration.
2. In the **Extractor**, **Applier**, and **InitialSync** lists, select the name of the task that you want to use as the default task and click **Save**.

   Note: You can select tasks for which you or another user is designated as the owner. The default value is "default," which indicates a standard task.

3. To test that the default task was reset, start the default task manually by selecting **Synchronize**, **Extract Data**, or **Apply to Target** from the **Run** menu or by clicking the corresponding button on the main toolbar.

Related Topics:

- "Running Replication Executables Manually from the Data Replication Console" on page 281

Editing Schedules

On the **Server Manager** tab > **Schedules** view, you can edit schedules for which you are designated as the owner. The idradmin user can edit all of the schedules.

You can change the schedule name and owner, associated configuration, number of days to retain schedule logs, schedule run options, and referenced tasks. You can also disable the schedule.

1. Click the **Server Manager** tab > **Schedules** view.
2. In the **Schedules** box, select the schedule that you want to edit.
3. Click the **Edit** icon button on the **Schedules** toolbar, or right-click the schedule name and click **Edit**.

   The **Editing** dialog box appears.
4. Edit any of the available fields.
5. Click **Save**.

Starting a Schedule Manually

A schedule typically starts based on the run options that are defined for the schedule. However, in certain situations, you might need to start a schedule manually.

For example, you might restart a schedule after stopping it or run a schedule again because one or more tasks failed.

You can start a schedule for which you are designated as the owner. The idradmin user can start any schedule.

1. On the **Server Manager** tab > **Schedules** view, select the schedule that you want to start.
2. Click the **Start** icon button on the Schedules toolbar, or right-click the schedule row and click **Start**.

Note: A schedule does not start if any running processes are blocking its tasks or if the Data Replication user is editing the configuration that is associated with the schedule in Edit mode. When a schedule fails to start, you can diagnose the problem by viewing a list of the processes that are blocking the scheduled tasks. On the **Server Manager** tab > **Schedules** view, select a task for which to view blocking processes. In the **Processes** box, select a replication process and click the **Blocking Information** icon button.

Stopping a Schedule Manually

A schedule stops based on the scheduling options that you set. However, you might need to stop a running schedule manually, for example, to add a new task to it.

You can stop a schedule for which you are designated as the owner. The idradmin user can stop any schedule.

1. On the **Server Manager** tab > **Schedules** view, select the schedule that you want to stop.

2. Click the **Stop** icon button on the Schedules toolbar, or right-click the schedule row and click **Stop Schedule**.

   Before stopping, the schedule processes complete normally.

   Note: In urgent situations in which you cannot wait for the processes to complete, click the **Abort** icon button on the Schedules toolbar, or right-click the schedule row and click **Abort Schedule** to force the processes to end.

Stopping and Restarting a Scheduled Task Manually

Scheduled InitialSync, Extractor, Send File, and Applier tasks stop when they complete processing. However, you can manually stop a running task or restart a stopped task that is part of a schedule if you are the owner of the task. The idradmin user can stop or restart any scheduled task.

1. On the **Server Manager** tab > **Schedules** view, select a task that has the status of running under a schedule node.
2. Click the **Stop** icon button on the Schedules toolbar, or right-click the task row and click **Stop Task**.

   Before stopping, the task completes normal processing.

   Note: In urgent situations in which you cannot wait for a task to complete processing, click the **Abort** icon button on the Schedules toolbar, or right-click the task row and click **Abort Task** to force the task to end.
3. If you need to restart a stopped task, click the **Start** icon button on the Schedules toolbar, or right-click the task row and click **Start Task**.

Starting a Task Manually from the Tasks View

You can start a previously defined task outside a schedule. If a task requires a configuration, you can run it for the configuration for which you are designated as the owner.

1. Click the **Server Manager** tab > **Tasks** view.
2. In the **Task and Task Dependencies** box, select the task that you want to start.
3. Right-click the task row and click **Start with Config** and then select a configuration to start a task for a particular configuration, or click **Start** if the task does not require a configuration.

   The **Task: <task_name> Configuration: <configuration_name>** window appears.

   Note: If you start a task that has a dependency on one or more other tasks, the other tasks are also started.
4. When the task completes, click **Close**.

Starting a Task Manually from the Configs View

For configurations for which you are designated as the owner, you can start a previously defined task outside a schedule.

1. Click the **Server Manager** tab > **Configs** view.
2. In the **Replication Configurations** list, select the configuration for which you want to start a task.
3. Right-click the configuration row and click **Start Task** and then select the task to start.

   The **Task: <task_name> Configuration: <configuration_name>** window appears.

   Note: If you start a task that has a dependency on one or more other tasks, the other tasks are also started.

4. When the task completes, click **Close**.

CHAPTER 13

Implementing Advanced Replication Topologies

This chapter includes the following topics:

Advanced Replication Topologies

In your environment, you might need to use more advanced and specialized replication topologies and modes to meet your replication requirements.

Data Replication supports the following advanced replication topologies:

- Continuous replication
- Replication from one source to multiple targets
- Bidirectional replication
- Cascade replication

Data Replication provides *loopback avoidance* for bidirectional replication and cascade replication and provides *conflict resolution* for bidirectional replication and replication from multiple sources to a single target. To accurately replicate change data, review the topics on these features and follow the configuration steps for the replication topology you need to perform.

Related Topics:

- "Loopback Avoidance for Replicated Data" on page 310
- "Conflict Resolution for Replicated Data" on page 41

Configuring Continuous Replication

With the Server Manager, you can perform continuous data replication. Continuous replication delivers source table changes to the target with very low latency.

Before you configure continuous replication, verify that Data Replication supports change data capture from online logs for the source database. Also, from the Data Replication Console, define the tasks that you want to execute continuously.

1. Load or create a configuration.
2. Switch to Edit mode.
3. On the **Extract Range** tab, select one of the following options if you want the Extractor to capture data from online logs:
   - For Microsoft SQL Server sources, select **Read from online transaction logs**.
   - For Oracle sources, select **Read from online redo logs**.
4. Click the **Runtime Settings** tab > **General** view.
5. In the **Continuous replication latency** field, enter the latency, in seconds, of change capture processing during continuous data replication.

   This value determines the duration of an Extractor microcycle. During a microcycle, the Extractor captures changes from database logs, creates an intermediate file, writes changes to the intermediate file, and then sleeps. When the latency period ends, the Extractor begins another microcycle.

   When setting the latency period, consider how it affects replication performance:

   - If the latency period is greater than the time it takes the Extractor to read changes from the database logs and write them to an intermediate file, the Extractor sleeps until the end of the latency period. To minimize inactive Extractor sleep periods, set the latency to a low value.
   - If the latency period is equal to or less than the time it takes the Extractor to read changes from the database logs and write them to an intermediate file, the Extractor continues processing and does not sleep. The Extractor begins a new microcycle as soon as it closes the intermediate file. However, if the latency is too low, performance of a long-running Extractor task might be degraded because of high disk memory or core loads. In this case, try increasing the latency value in sub-second increments such as 0.1 seconds.

   Default is 0.333 seconds, which is usually acceptable to process all changes in a microcycle.

   Note: After the Extractor creates the intermediate file, the Applier can start reading changes and applying them to the target. Therefore, the Applier might also run during a microcycle. For large transactions with many changes, the Applier waits until the Extractor writes all of the changes to the intermediate file, which might take several microcycles, and then begins apply processing.
6. Save the configuration.
7. On the **Server Manager** tab > **Schedules** view, create a schedule with the tasks you defined for continuous replication.

   Important: Do not add subtasks that have dependencies on other tasks.

   To create a schedule, use one of the following methods:

   - Click **New** to define a schedule in the **New** dialog box. In the **Run** field, you must select **Continuous**. Also define a run period under **Schedule Options**. In the **End date** field, you can specify a date with a year up to 2037.
   - Click **Wizard** to start the **Schedule wizard**. In Step 1, you must select **Continuous** as the execution scheme. In Step 2, define a run period under **Schedule Options**.

Related Topics:

- "Scheduling Replication Tasks" on page 281

Configuring Data Replication from One Source to Multiple Targets

You can use the Server Manager with the InitialSync, Extractor, and Applier components to replicate data from a single source to multiple targets.

Note: Data Replication does not support replicating data from a single source to multiple Amazon Redshift or Apache Kafka targets.

One of the target databases is considered to be the *primary target database*. The other *secondary target databases* must be the same target type as the primary target. Also, all of the mapped target tables must have identical schema definitions. However, the target schemas can have different names.

All of the mapped columns in the secondary target tables must have the same names as the corresponding columns in the primary target tables and have datatypes that are compatible with those of the corresponding columns in the primary target tables.

When you configure replication, define one primary target and multiple secondary targets. The primary target is the target database that you defined on the **Target Database** tab.

To configure and run the replication job, complete the following high-level tasks from the Data Replication Console:

1. Define a Server Manager Main server on a machine in your environment and connect to it. The Main server does not need to run on the source or primary target.
2. Define a subserver for each of the other target machines.
3. From the **Server Manager** tab > **Connections** view, define a connection for the primary target database and a connection for each secondary target database.
4. Create and save the configuration. On the **Target Database** tab, specify the connection that you defined for the primary target database.
5. On the **Routing** tab, define each target server and connection. Optionally, perform the following tasks:
   - Filter rows by table so that different targets receive different subsets of rows.
   - Exclude one or more entire tables from replication to a particular target.
   - Rename the target schema so that each target uses a different schema name.
   - Override the apply mode or audit log table schema for a particular target.
6. Run the replication.

Related Topics:

- "Replication from a Single Source to Multiple Targets" on page 24

Defining Connections for Multiple Targets

To replicate data to multiple target databases, define a connection for the primary target database and a connection for each secondary target database. For the source database, you can either define a connection or use a pre-existing one.

1. On the **Server Manager** tab > **Connections** view, click the **New** icon button on the Connections toolbar.

   The **New** dialog box appears.
2. In the **Name** field, enter a name for the connection.

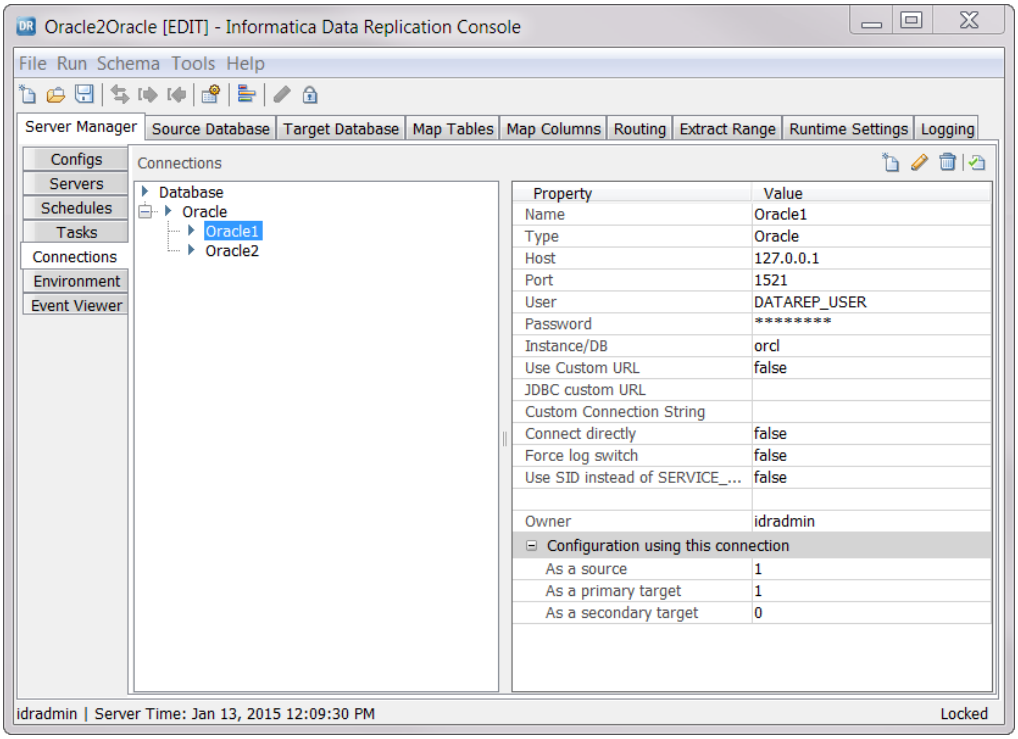
   Note: Connection names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates connection names that are longer than 100 characters.
3. In the **Owner** list, select an owner for the connection.
4. In the **Type** list, select the target database type.
5. On the **Database** tab, enter connection information for the target, including the host name or IP address, port number, user name and password, and instance or database name.

   In an Oracle ASM environment, enter connection information on the **ASM Settings** tab.
6. To test the connection, click **Test Connection**.

   If the connection is successful, "Connected" is displayed to the left of the **Test Connection** button.

7. Click **OK**.

   The **Connections** view lists the connection. For example, the following image lists the Oracle1 connectiont:



   If you select a listed connection, the connection details are displayed on the right. Under **Configuration using this connection**, you can view the number of source databases, primary targets, and secondary targets that use the connection definition.

   Tip: To edit a database connection, select it in the **Connections** list and then click the **Edit** icon button on the Connections toolbar. In the **Editing** dialog box, enter changes and click **Save**.

Creating a Configuration for Replicating Data to Multiple Targets

From the Informatica Data Replication Console, you can create a configuration that replicates data to multiple targets.

Note: Data Replication does not support replicating data from a single source to multiple Amazon Redshift or Apache Kafka targets.

On the **Target Database** tab, select the connection that you previously defined for the primary target on the **Server Manager** tab > **Connections** view. Then add the secondary targets on the **Routing** tab.

Instead of creating a new configuration, you can modify an existing configuration file that has a single target. In this case, you must rename the target schema for each secondary target to a name that matches the schema name of the primary target.

1. In the **Server Manager** tab > **Configs** view, click **New** on the main toolbar.
2. On the **Source Database** tab, connect to the source database.
3. On the **Target Database** tab, in the **Connections** list, select the connection that you defined for the primary target.
4. Map tables and columns, set the extract range, and enter runtime and logging settings, as usual.

5. Save the configuration and switch to **Edit** mode.

   The **Routing** tab becomes available.

6. On the **Routing** tab, specify the Server Manager subserver and connection for a secondary target:

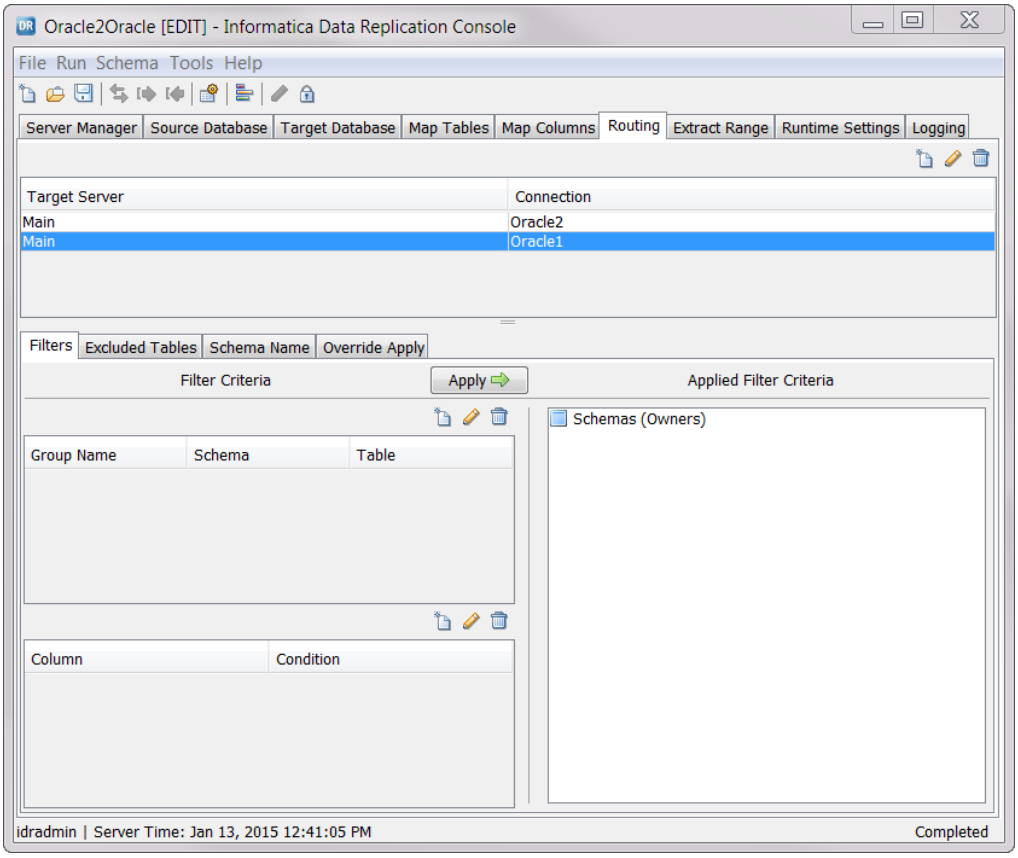
   a. Click **New**.

   b. In the **New** dialog box, select the Server Manager subserver and database connection that you previously defined for the target.

   c. Click **OK**.

   The target appears in the upper list box on the **Routing** tab.

   The following image shows the **Routing** tab with the secondary target:



   If the target type uses a recovery table, the Data Replication Console prompts you to specify the recovery table name and schema.

7. To define a recovery table for a target, enter the schema name and recovery table name in the appropriate fields. Then click **OK**.

   When you start the Applier later, it generates the recovery table on the target.

   Important: Create a separate recovery table for each target.

8. Repeat Step 6 for each secondary target. For each secondary target that uses a recovery table, also repeat Step 7.

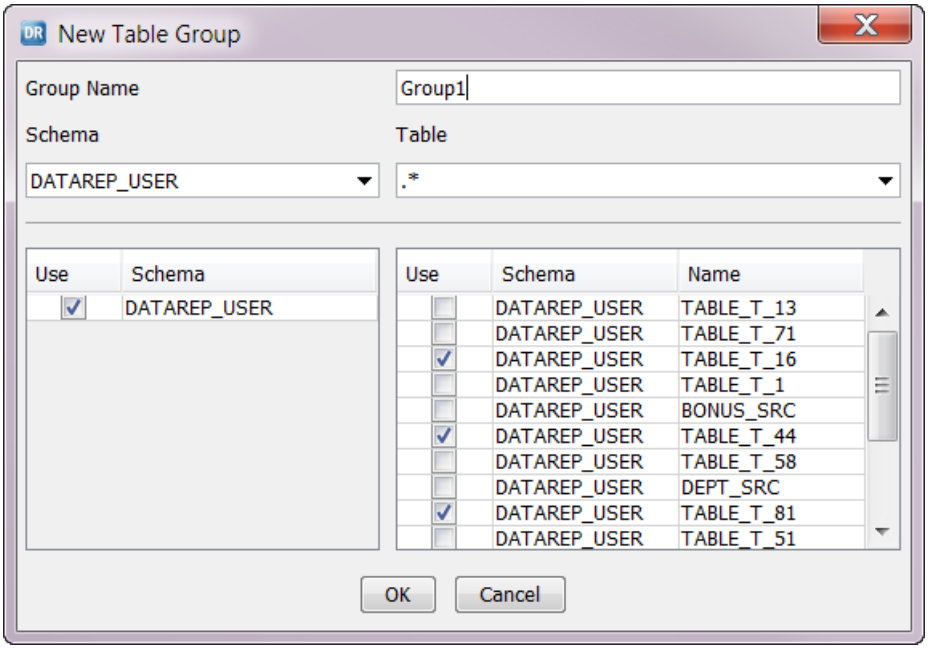
From the subtabs on the **Routing** tab, you can filter table rows, exclude tables, and override the apply mode and log table schema for a secondary target.

Filtering Table Rows for Selective Replication to Different Targets

By default, Data Replication replicates data from source tables to corresponding target tables on all of the target servers. However, you can define a filter that selects a subset of table rows to replicate to a particular target.
First define the group of source tables to filter, and then define the filter conditions for the table group.

1. Verify that the configuration is in Edit mode.
2. On the **Routing** tab, select a target server.
3. On the **Filters** subtab, click the **Add** icon button on the upper toolbar to add a group of tables.

   The **New Table Group** dialog box appears.



4. In the **Group Name** field, enter a name for the table group.

   Note: Table group names can contain only the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates table group names that are longer than 100 characters.

5. In the **Schema** field, select a schema name or enter a mask for the schema name. A mask contains the asterisk (*) wildcard.

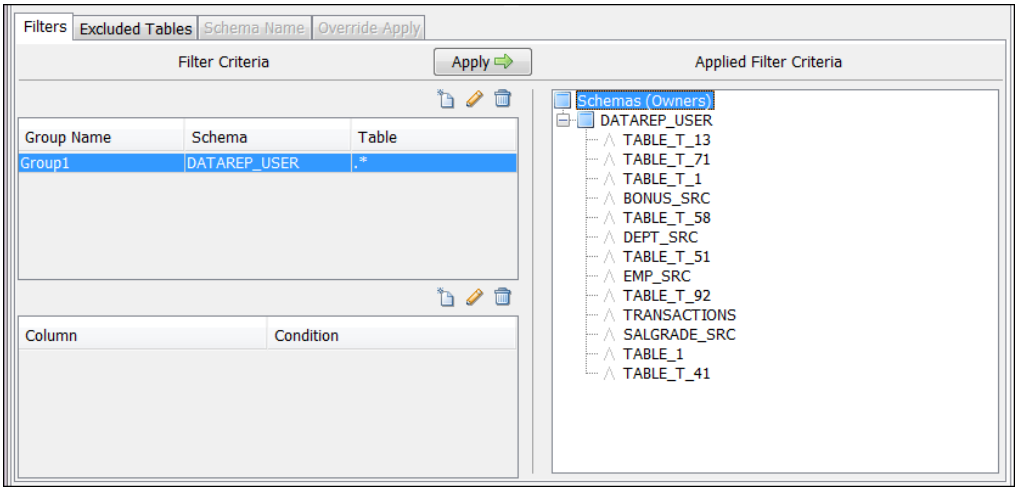
   The lower left box lists the schemas that match your entry.

6. In the **Table** field, select a table name or enter a mask for the table name. A mask contains the asterisk (*) wildcard.

   The lower right box lists the tables in the selected schema or schemas that match your table name entry.

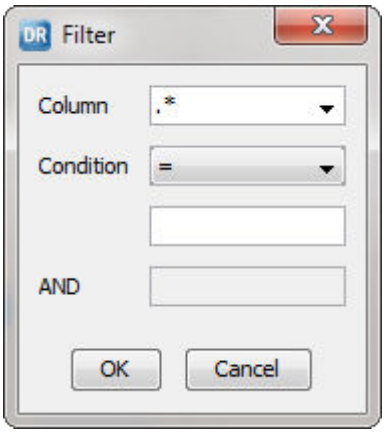
7. In the lower list boxes, select the schemas and tables to which the filter will apply. Then click **OK**.

The **Filters** subtab displays the group name. The following image shows the **Filters** subtab with a new group name selected:



8. To add a filter condition, select the group name in the upper list box. Then click the **Add** icon button above the lower list box for columns and conditions.

   The **Filter** dialog box appears.



9. To define a filter condition, perform the following steps:
   a. In the **Column** field, define a column name mask to select the columns that are part of the filter.

      Tip: You can enter only the asterisk (*) wildcard to include all columns.

   b. Select a condition operator.

   c. If the operator type requires a value to form the filter condition, enter the value in the field below the operator list. For example, if the operator is >, enter 1 to form the condition “greater than 1.”

      Tip: For date and time columns, use the following format for the value: `yyyy-mm-dd hh:mm:ss`.

   d. If the operator type requires a second value, enter a value in the field after AND. For example, if the operator is BETWEEN, enter 5 before AND and 10 after AND to form the condition “between 5 and 10.”

   e. Click **OK**.

      The filter condition appears on the **Filters** subtab. The Data Replication Console prompts you to apply the filter condition.

10. Click **Yes** to apply the filter condition immediately. Click **No** if you want to apply the condition manually later.

    To apply the filter condition manually, click **Apply**. Then click **Yes** at the confirmation prompt.

    The right side of the **Filters** subtab shows the filter hierarchy. The subtab lists all of the tables and columns that match the filter criteria and the condition that is applied to each filtered column.

11. Repeat steps 1 through 8 for other target servers, as needed.

Notes:

- For Oracle sources, InitialSync creates a view for internal use when materializing targets with DBLinks. To filter data for a particular target, InitialSync executes SELECT statements with a WHERE clause for the view. To generate valid SELECT statements, set the where_condition_surround_table_and_col_names_with_special_chars parameter to **true** in the *DataReplication_installation*/uiconf/default.cfg file. If the file does not exist, use a text editor to create the file in the uiconf directory and then add the parameter in the file. For more information, see "Default.cfg File" on page 391.
- For each Oracle source table in a group, after you apply a filter condition, the Data Replication Console creates a virtual index that includes the columns for which the filter condition is defined, if a virtual index does not already exist. The virtual index is required to enable supplemental logging for the source tables. To name the virtual index, the Data Replication Console uses the table name followed by the _FILTER suffix.
- If you create a filter for a particular target, Data Replication skips the source table rows that do not match the filter criteria when replicating changes to this target. If you later update at least one of the columns in a source table row that was previously filtered out for this target to a value that no longer meets the filter criteria, the source table row will be replicated to the target. However, for Oracle sources, any columns that are not in a supplemental log group created by Data Replication will have a null value in the replicated row on the target.
- Data Replication applies the filters that you defined from the **Filters** subtab when the Send File task runs. If you add a filter after the Send File task ends, Data Replication does not apply the new filter to the replicated data.

Related Topics:

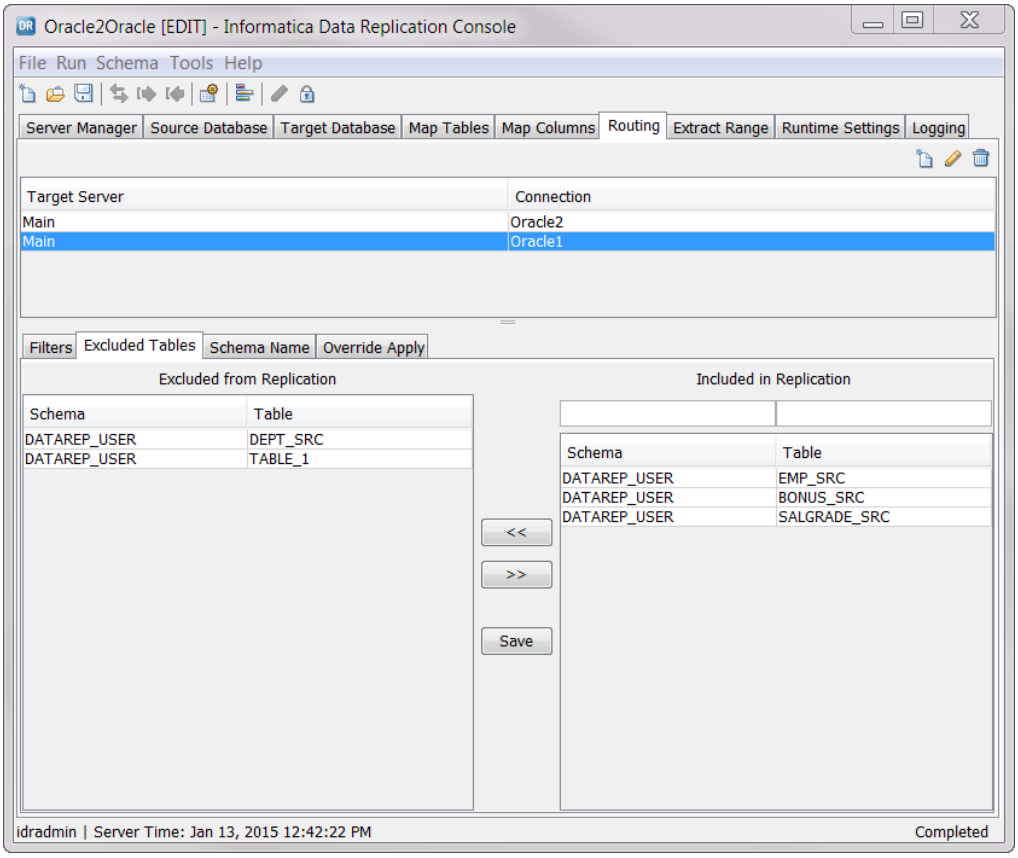
- "Managing Open Transactions" on page 336
- "Updating Source and Target Metadata for a Configuration in Non-interactive Mode" on page 453

Excluding Entire Tables from Replication to a Target Server

You can exclude one or more entire tables from replication to a particular target server.

1. On the **Routing** tab, select the target server and then click the **Excluded Tables** subtab.

   The following image shows this subtab:



2. In the **Included in Replication** box, select the tables that you want to exclude.
3. Click the left arrow button to move them to the **Excluded from Replication** box.
4. Click **Save**.

Note: Data Replication excludes the tables that you specified on the **Excluded Tables** subtab when the Send File task runs. If you add a filter after the Send File task ends, Data Replication does not apply the new filter to the replicated data.

Renaming the Target Schema

For replication of data from one source to multiple targets, the schema of all targets must have an identical structure. However, the schema names of the secondary targets that you specify on the **Routing** tab can be different from the schema name of the primary target that you specify on the **Map Tables** tab.

If you replicate change data to a table with a specific schema name on the primary target and want to route the changes to the corresponding table on a secondary target using a different schema name, you must, for the secondary target, override the primary target schema name table with the secondary target schema name. Otherwise, the Applier ends abnormally because it cannot find the schemas of the secondary targets.

For example, if you map a target table that uses the schema name of *Schema1* on a primary target and the schema name of *Schema2* on a secondary target, you must override *Schema1* with *Schema2* on the secondary target.

Note: This task does not apply to Netezza targets. For Netezza targets, Data Replication does not support overriding the target schema names because a Netezza user can own only one schema. For Netezza secondary targets, Data Replication replicates changes to the target schema that corresponds to the user specified in the connection settings for the target.

1. On the **Routing** tab, select a secondary target server and then click the **Schema Name** subtab.
2. Click **New**.

   The **New** dialog box appears.
3. Select a primary target schema that is defined in the configuration and enter a new schema name.

   Note: For databases that use case-sensitive database object names, enter the schema name in the same case that the database uses. For Oracle unquoted identifiers, use uppercase.
4. Click **OK**.

Overriding the Applier Settings for a Target

When you configure a replication job that has multiple targets, you can override some Applier settings for a target.

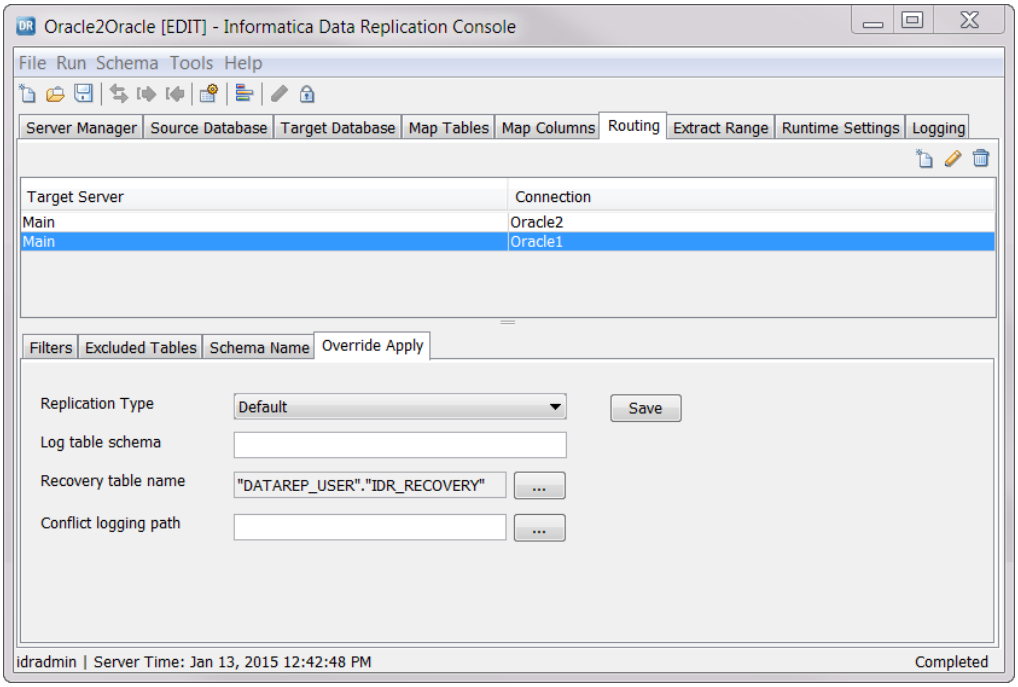
The following types of settings can be overridden for a target:

- The apply mode
- The schema name for Merge Apply audit log tables on the target
- The name of the recovery table in the target database

- The location of the log file that is used for conflict resolution

1. On the **Routing** tab, select the target server for which to override settings and click the **Override Apply** subtab.

   The following image shows this subtab:



2. To select another apply mode for the selected target, click one of the following options In the **Replication Type** list:

   - **SQL Apply**
   - **Audit Apply**
   - **Merge Apply**

   Note: You can use Audit Apply mode for a secondary target to add audit log tables in the primary target database that use the same schema as the regular target tables in the same database. In this case, the primary target and secondary target use database connections with the same connection information. To create valid SQL statements for the audit log tables for the secondary targets, set the apply.force_log_table_suffix_for_audit advanced runtime parameter to 1. Also, ensure that the names of the audit log tables end with the suffix that is specified in the **Log table suffix for merge apply** field on the **Runtime settings** tab > **Calculated Columns** view.

3. To change the schema name for audit log tables on the target, enter a schema name in the **Log table schema** box.

4. To change the recovery table name, click the **Browse** button for the **Recovery table name** field.

   Then, enter the database, schema, and recovery table name in the appropriate fields and click **Yes**.

   Note: For Cloudera, Flat File, and Hortonworks targets, skip this step. These target types do not use a recovery table.

   When you later start the Applier, it generates the recovery table on the target.

5. To override the path to the log file that the Applier uses for conflict resolution, click the **Browse** button for the **Conflict logging path** field.

   The Applier logs data conflicts that occur during apply processing to this log file.

6. Click **Save**.

Running a Replication Job with Multiple Targets

To run a replication job with multiple targets, you can schedule it from the **Server Manager** tab > **Schedules** view or run the InitialSync, Extractor, and Applier manually.
To run the tasks manually from the Data Replication Console, perform the following actions:

- To run InitialSync, click the **Synchronize** toolbar button or click **Run** > **Synchronize** on the menu bar. In the **Select Targets** dialog box, select all targets for which to perform InitialSync processing and click **OK**.
- To run the Extractor, click the **Extract Data** toolbar button or click **Run** > **Extract Data** on the menu bar.
- To run the Applier, click the **Apply to Target** toolbar button or click **Run** > **Apply to Target** on the menu bar. In the **Select Targets** dialog box, select all targets for which to apply changes and click **OK**.

Configuring Bidirectional Replication

To replicate changes between two systems in both directions simultaneously, you must create two configurations, one for each direction. Also configure loopback avoidance and conflict resolution rules for the replicated data.

Note: Data Replication does not support bidirectional replication for configurations that have a MySQL source or an Apache Kafka target.

All of the tables in one of the databases must be initially empty. Before you start bidirectional change replication, run InitialSync to materialize the empty tables. Use the configuration that specifies the database with the empty tables as the target.

In the following steps, *Database A* and *Database B* represent two databases that run on separate systems. All of the tables in Database B are initially empty. After you start replication, loopback avoidance prevents the Database A changes that were applied to Database B from being returned to Database A.

Note: Data Replication does not replicate DDL changes for bidirectional replication.

1. Create a replication configuration with Database A as the source and Database B as the target.

   Ensure that you disable DDL replication on the **Map Tables** and **Map Columns** tabs.
2. Define conflict resolution rules for the replication configuration. For more information, see "Configuring Conflict Resolution" on page 233.
3. Run the InitialSync task to materialize the empty tables in Database B.
4. Create a configuration for the reverse replication of data, which has Database B as the source and Database A as the target.
   a. Click **File** > **Generate Reverse Configuration**.

      The Data Replication Console generates a configuration and opens it in Edit mode.
   b. Configure replication settings on the **Extract Range** tab and the **Runtime Settings** tab > **Advanced Settings** view.
   c. On the **Map Tables** tab, click **Configure Start Point**. Under **Applier start point**, select all of the mapped tables and click **Get SCN From Source** to use the current SCN or LSN value on the source as the Applier starting point. Click **Save**.

d. Click **Save** on the toolbar.

The Data Replication Console applies the conflict resolution rules that you defined for the first configuration to the generated reverse-replication configuration.

5. For the reverse-replication configuration, create a recovery table on Database A. On the **Runtime Settings** tab > **Advanced Settings** view, double-click the apply.recovery_table parameter row. Then specify a full table name that includes the schema or owner name, and click **OK**.

   When you start the Applier later, it creates a recovery table with the specified name in Database A.

6. On the **Server Manager** tab > **Schedules** view, create schedules for both configurations. In each schedule, include the Extractor, Send File, and Applier tasks.

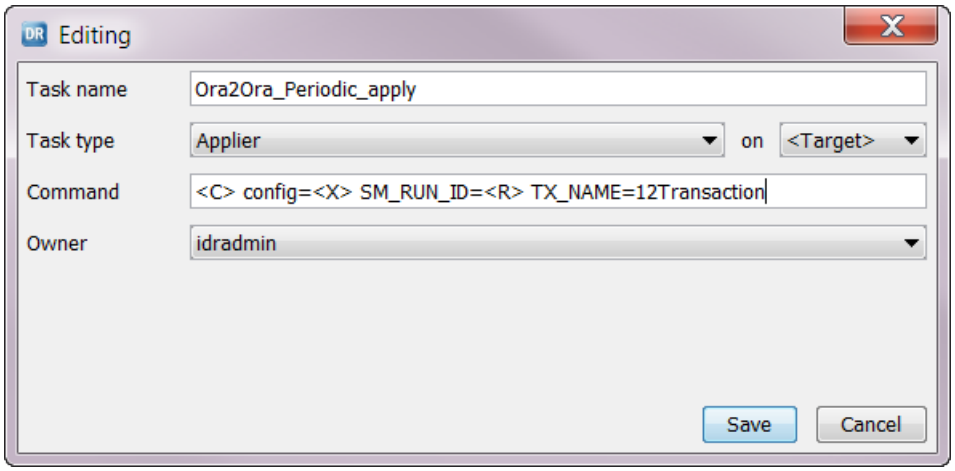
7. Optionally, configure the Applier task for one or both configurations to use transactions that have a transaction name other than the default name to support loopback avoidance.

   a. On the **Server Manager** tab > **Tasks** view, select the Applier task row and click **Edit**.

      The **Editing** dialog box appears.

   b. In the **Command** field, append the TX_NAME parameter. As the parameter value, specify the transaction name.

      The following image shows example Applier settings:



   c. Click **Save**.

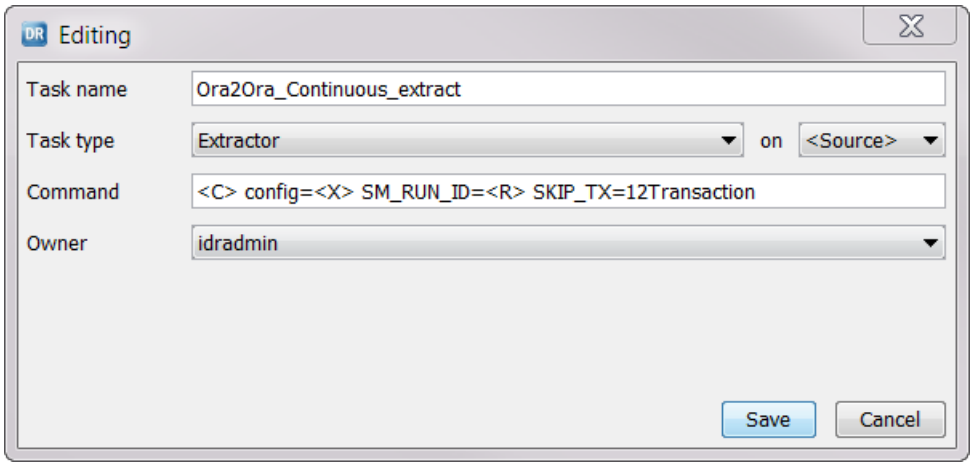
8. If you configured the Applier task to use transactions that have a name other than the default name, configure the Extractor task on the same system to skip transactions with names that match the TX_NAME name value you specified when capturing data for reverse replication.

   a. On the **Server Manager** tab > **Tasks** view, select the Extractor task row for the reverse-replication configuration and click **Edit**.

      The **Editing** dialog box appears.

   b. In the **Command** field, append the SKIP_TX parameter. As the parameter value, specify the transaction name that you want to skip.

The following image shows example Extractor settings:



c. Click **Save**.

9. Run the replication schedules for each configuration.

Related Topics:

- "Bidirectional Replication" on page 26
- "Conflict Resolution for Replicated Data" on page 41
- "Loopback Avoidance for Replicated Data" on page 310
- "Generating a Reverse-Replication Configuration" on page 362

Configuring Cascade Replication

For a cascade replication that replicates data unidirectionally across a chain of databases, create a configuration for each pair of databases in the chain.

Note: Data Replication does not support cascade replication for configurations that have an Apache Kafka target.

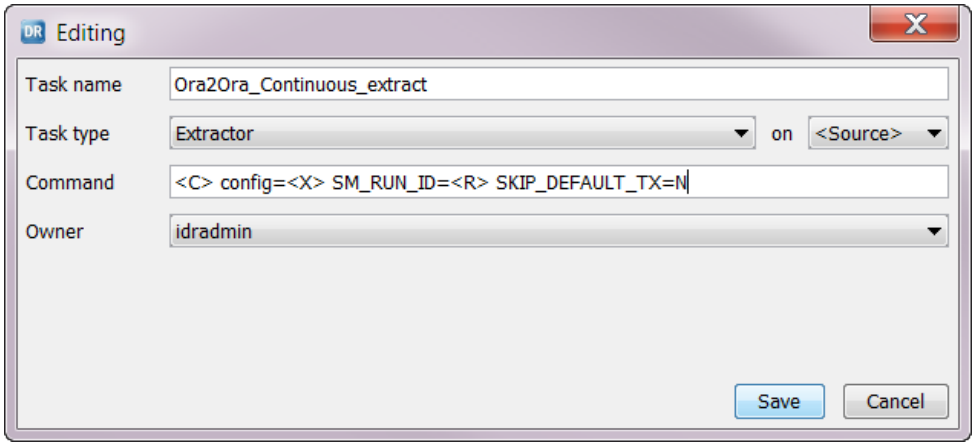
1. Create a replication configuration for each pair of databases in a chain.
2. For each configuration, run InitialSync. Begin with the last configuration in the cascade replication and end with the first configuration in the cascade replication.
3. On the **Server Manager** tab > **Schedules** view, create a schedule for each configuration.
4. On the **Server Manager** tab > **Tasks** view, perform the following steps for each Extractor task in the configurations in the cascade replication, beginning with the second configuration:

   Note: Do not perform these steps for the first configuration in the cascade replication.

   a. Right-click the Extractor task in the **Name** column of the Tasks and Tasks Dependencies list and click **Edit**, or select the Extractor task in the **Name** column and click the **Edit** icon button in the Tasks and Task Dependencies toolbar.

      The **Editing** dialog box appears.

b. In the **Command** field, enter SKIP_DEFAULT_TX=N to disable loopback avoidance. The following image shows this view:



If you do not want to disable loopback avoidance, use the default setting of SKIP_DEFAULT_TX=Y. This setting replicates only the changes that originate on the database immediately before the current database in the cascade replication and skips the changes that originate on preceding databases in the cascade replication. For example, when replicating changes from Database B to Database C, Data Replication skips the transactions that originated on Database A and replicates only the changes that originated on Database B.

c. Click **Save**.

d. Repeat these steps for the remaining configurations in the cascade replication that contain Extractor tasks.

5. Run the replication schedule for each configuration.

   From any database in the chain, Data Replication replicates forward all the changes that originally occurred on that database, unless you specified SKIP_DEFAULT_TX=Y in step 4.

Related Topics:

- "Cascade Replication" on page 26
- "Loopback Avoidance for Replicated Data" on page 310

Loopback Avoidance for Replicated Data

Data Replication provides loopback avoidance to support bidirectional replication topologies that use one or more databases as both a source and target simultaneously.

For bidirectional replication, loopback avoidance prevents Data Replication from replicating changes back to the database from which they were originally captured.

Data Replication supports loopback avoidance by default. InitialSync and the Applier use transactions that have the default name of DbSyncTransaction to replicate data to a target. The Extractor uses this transaction name to distinguish the transactions that originally occurred on the database from the transactions that InitialSync and the Applier replicated. By default, the Extractor does not capture data from transactions that have the default transaction name of DbSyncTransaction. If you want the Extractor to capture change data from transactions that have this default name, set the SKIP_DEFAULT_TX command line parameter to N.

Note: For cascade replication configurations, always set the SKIP_DEFAULT_TX command line parameter to N to disable loopback avoidance.

For DB2, Data Replication uses an application name instead of a transaction name to distinguish the changes that originally occurred on the database from the changes that InitialSync and the Applier processed. The default application name is also DbSyncTransaction.

In a complicated replication topology, you might need to use transactions that have a name other than the default transaction name to replicate changes to a target. In this case, use the TX_NAME command line parameter to specify the transaction name that you want InitialSync and the Applier to use. Also, for the Extractor that captures changes from the same database, specify the SKIP_TX command line parameter to skip the transactions that have the TX_NAME name that you specify for InitialSync and the Applier.

For more information about the command line parameters for loopback avoidance, see Appendix D, "Command Line Parameters for Data Replication Components" on page 438.

Related Topics:

- "Bidirectional Replication" on page 26
- "Conflict Resolution for Replicated Data" on page 41
- "Configuring Bidirectional Replication" on page 307
- "Cascade Replication" on page 26
- "Configuring Cascade Replication" on page 309

CHAPTER 14

Monitoring Data Replication

This chapter includes the following topics:

Types of Monitoring Information

The Data Replication Console provides the following types of information for monitoring replication jobs and performance:

- Latency statistics for continuous replication
- Extractor performance statistics
- Statistics on intermediate files
- Content of the intermediate files
- Schedule execution logs
- Server Manager logs
- Open transactions on the source database

In the Console, you can also configure email and SNMP notifications for Data Replication events by subscriber and event type. Optionally, you can attach message log files to the notifications.

Note: You can also use the Server Manager Command Line Interface to view the execution logs of replication tasks, list open transactions on the source, and mark open transactions for commit or rollback processing on the target. For more information, see the *Informatica Data Replication Command Line Interface for the Server Manager*.

Replication Statistics

From the Informatica Data Replication Console, you can view detailed performance statistics for the Extractor, Applier, and Send File tasks. You can also view latency statistics for continuous replication and content of the intermediate files.

More specifically, you can get the following types of statistical information:

- View Extractor performance statistics for a specified period, including the amount of data processed in MB per second and number of transactions processed.
- View intermediate file performance statistics for a specified period, including the amount of data processed, in MB per second, for the data files and transaction files that comprise the intermediate files.
- View records that comprise a particular intermediate file and the before and after images for the operations.
- View the latency of the Extractor, Applier, and Send File tasks by heartbeat, where a heartbeat corresponds to an intermediate file.
- View the average latency of Extractor, Applier, and Send File processing during continuous replication over a specific time period, and total end-to-end latency.
- Drill down to run times of the Extractor, Applier, and Send File tasks for each intermediate file.
- Drill down to transaction statistics by table for each intermediate file.

You can use the latency statistics to analyze continuous replication performance, change volume, and composition.

Replication Statistics Window

While a replication job is running or after it completes, you can view performance statistics for the Extractor, Applier, and Send File tasks. You also can view latency statistics for the Extractor, Send File, and Applier tasks and the total end-to-end replication. The Server Manager collects statistics if the DisableStatistics property for the Server Manager is set to the default value of 0.

The **Statistics** window contains separate tabs for latency, extractor performance, and intermediate file. At the top of each tab, you can specify the period for which to display statistics. Also, each tab has a graph view and data view of the statistics. Click the **Graph** subtab to get a graphical representation of performance statistics. You can save the graph as an image to a disk. Click the **Data** subtab to view the statistics as numbers.

To open the **Statistics** window, use one of the following methods:

- Click the **Statistics** button on the toolbar while a configuration file or schedule is open or selected.
- On the **Server Manager** tab > **Configs** view, right-click a configuration file row and click **Statistics** or click **Statistics** on the toolbar.
- On the **Server Manager** tab > **Schedules** view, right-click a schedule that has an associated configuration file and click **Statistics**.

Before the **Statistics** window opens, a progress dialog box momentarily appears while the Server Manager collects statistics for the replication job. If this process takes a long time, you can cancel it from the progress dialog box.

Related Topics:

- "Editing Properties for the Main Server or a Subserver" on page 138

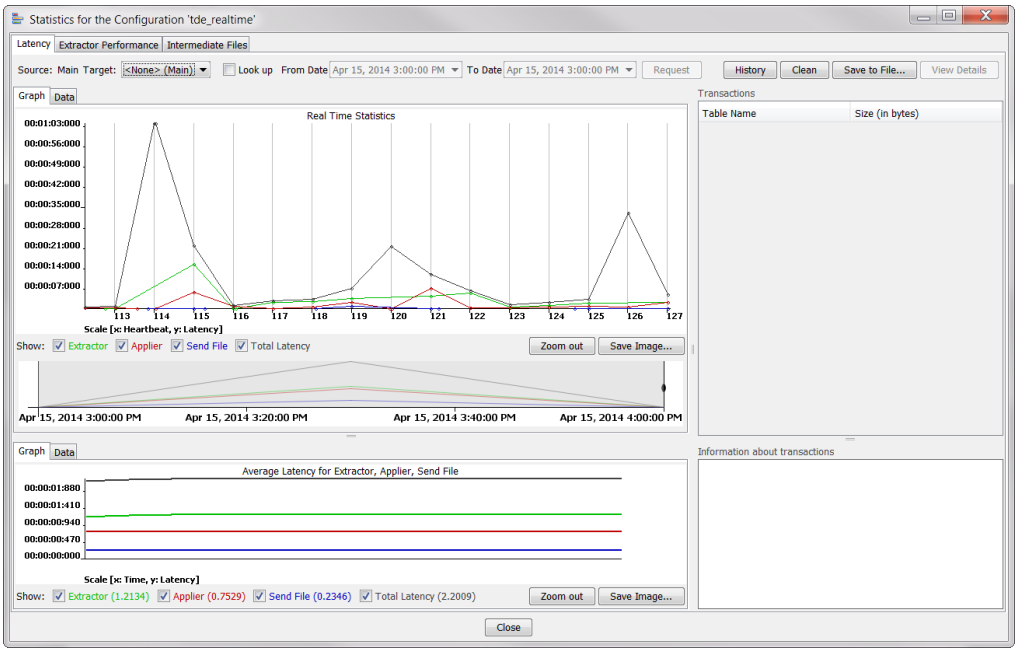
Viewing Latency Statistics for Continuous Replication

On the **Latency** tab, you can view the latency of the Extractor, Applier, and Send File tasks by heartbeat for a continuous replication.

For each intermediate file, you can view transaction-related statistics. You can also view the average performance of each replication task and the total end-to-end replication latency for the continuous replication.

1. In the **Statistics** window, click the **Latency** tab.

   The following image shows the **Graph** view of the **Latency** tab:



2. In the **Target** box, enter a target connection.
3. To view statistics for a specific time period, select **Lookup**. Then enter date and time values in the **From Date** and **To Date** fields.
4. Click **Request** to display statistics.
   - The upper box displays latency statistics for the Extractor, Applier, and Send File tasks in terms of intermediate files. In the graph, a heartbeat corresponds to an intermediate file. The graph also shows the total latency for the Send File end-to-end replication.
     The **Data** view is a numeric representation of the graphed data. This view also includes the start and end times for the Extractor and Applier.
   - The middle box displays latency statistics for the Extractor, Applier, and Send File tasks and the total latency for the end-to-end replication aggregated by a week. You can move the time mark to get detailed statistics in the upper box for a specific period of time.
   - The lower box provides the average latency across all of the intermediate files. The **Graph** and **Data** views show the average latency for the Extractor, Applier, and Send File tasks and the end-to end replication. The **Data** view also shows the average total time and absolute total time for end-to-end replication.

Tips:

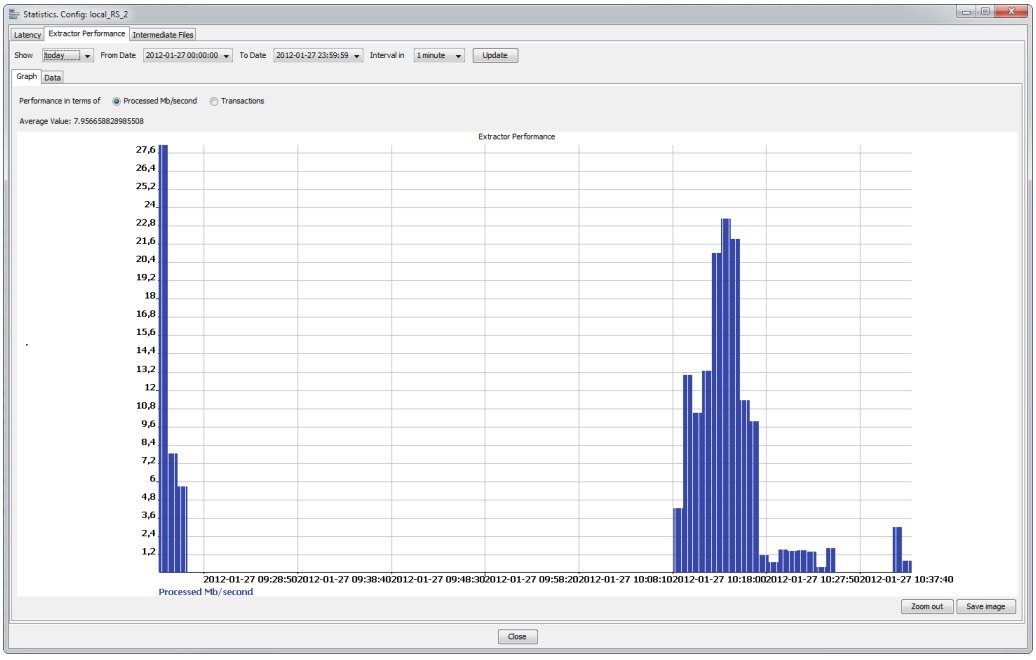
- Click **History** to display a historical statistics graph similar to the upper one on the **Latency** tab but with additional aggregation by time. You can aggregate by seconds, minutes, hours, days, and weeks.
- Click **Clean** to clear the graph contents to start again.
- Click **Save to File** to save the table in the upper **Data** subtab to a text file.

Viewing Extractor Performance Statistics

On the **Extractor Performance** tab, you can view statistics that indicate Extractor performance, including the amount of data processed in MB per second and number of transactions processed. The **Graph** subtab provides a graphical representation of the statistical data. The **Data** subtab provides numeric statistics.

1. In the **Statistics** window, click the **Extractor Performance** tab.
2. To specify a period for which to show statistics, complete one of the following fields:
   - In the **Show** list, select today, yesterday, or last week.
   - In the **From Date** and **To Date** fields, enter a date and time to define a specific period for which to display statistics.
3. In the **Interval in** field, select 5 seconds, 1 minute, 1 hour, or 1 day to define the interval at which the statistics are reported and graphed.
4. On the **Graph** subtan, click **Processed MB/sec** or **Transactions** to indicate whether to show performance statistics in terms of the amount of data or the number of transactions processed.
5. Click **Update** to display statistics.

   The following image shows the **Graph** view of the **Extractor Performance** tab, with the statistics in MB processed per second:

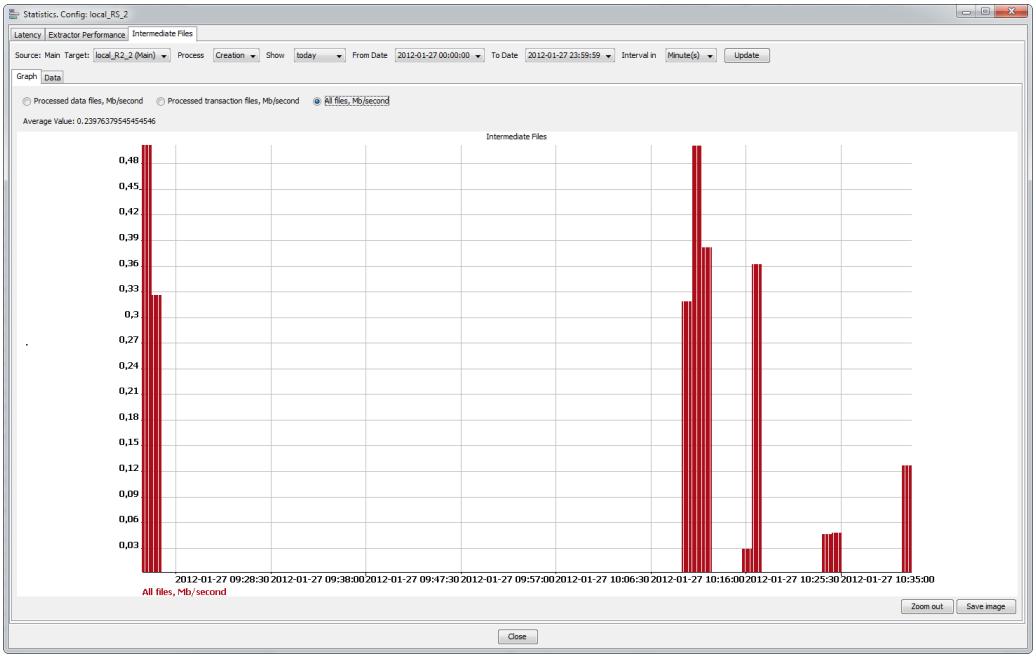


Viewing Statistics for Intermediate Files

On the **Intermediate Files** tab, you can view statistics on the intermediate files processed by the Extractor or Applier. The Data Replication Console displays statistics for the data files and transaction files that comprise the intermediate files.

1. In the **Statistics** window, click the **Intermediate Files** tab.
2. In the **Target** box, select the target connection.
3. In the **Process** field, select one of the following options:
   - **Creation**. Show statistics based on when each intermediate file is copied to the target.
   - **Applying**. Show statistics based on when each intermediate file is processed by the Applier.
4. In the **Show**, **From Date**, and **To Date** fields, define the period for which to show statistics.
5. In the **Interval In** field, select 5 seconds, 1 minute, 1 hour, or 1 day to define the interval at which the statistics are reported and graphed.
6. On the **Graph** subtab, select one of the following options to indicate whether to show MB processed per second for the data files, the transaction files, or both of these file types:
   - **Processed data files**
   - **Processed transaction files**
   - **All files**, which means both the data and transaction files
7. Click **Update** to display the statistics.

   The following image shows the **Intermediate Files** tab for both the data and transaction files that comprise the intermediate files:



Intermediate Files

In the Data Replication Console, you can drill down from a replication configuration to view the associated intermediate files. The idradmin user can view the content of intermediate files for all configurations. Regular users can view the content of intermediate files for the configurations that they own. Also, if you know the intermediate file name, you can view the file directly, without drilling down from the configuration. This feature is primarily for use by the Data Replication administrator and Informatica Global Customer Support.

You can view the following types of items in intermediate files, even if the files are encrypted:

- A list of records in the file
- The records of a particular transaction
- The before and after images for each record in a .dat file

Optionally, you can filter the records in an intermediate file by record type, table, or date and time.

Note: After the Server Manager removes a processed intermediate file, the file contents are no longer available for viewing in the Data Replication Console.

Related Topics:

- "Viewing the Intermediate Files for a Configuration" on page 317
- "Viewing a Specific Intermediate File Directly" on page 319
- "Change Record Information in Intermediate Files" on page 321

Viewing the Intermediate Files for a Configuration

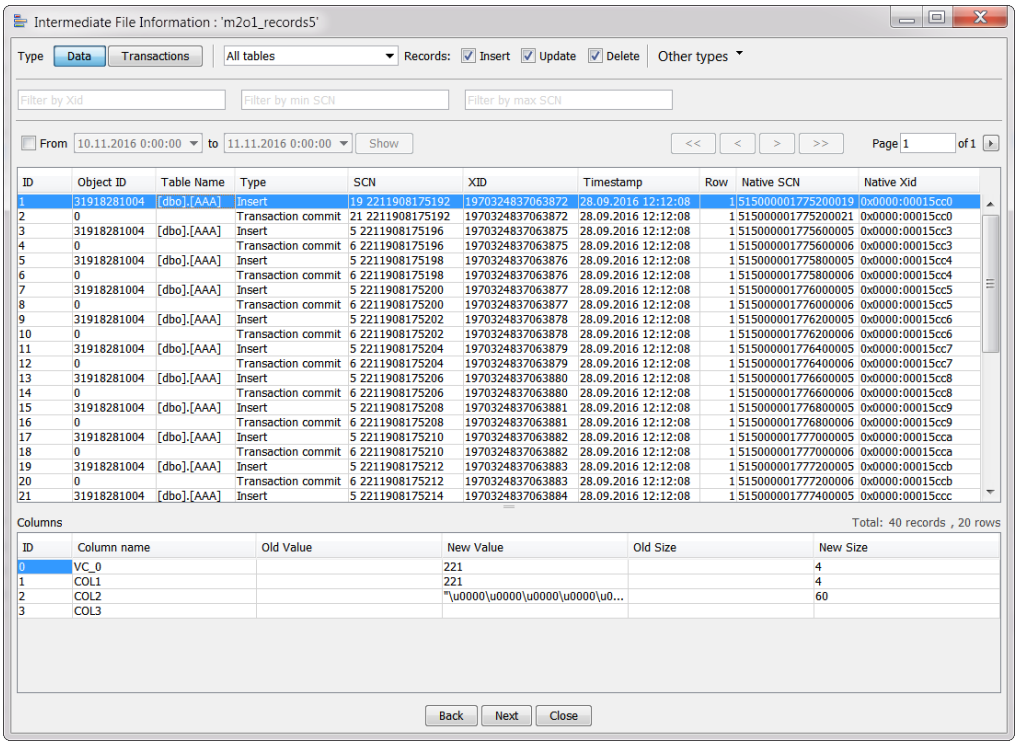
In the Data Replication Console, you can view the content of the .dat and .trn files that comprise the intermediate files for the configurations that you own. The idradmin user can view the intermediate files for all configurations.

1. On the **Server Manager** tab > **Configs** view, select a configuration for which to view intermediate files.
2. To view a list of the intermediate files for the configuration, use one of the following methods:
   - Click the **Statistics** button on the Replication Configurations toolbar, or right-click the configuration row and click **Statistics**. In the **Statistics for the Configuration <config_name>** window, click the **Latency** tab. In the upper left, click **Data**.
   - Click the **Intermediate Files** button on the Replication Configurations toolbar, or right-click the configuration row and click **Intermediate Files**. The **Intermediate Files for the Configuration <config_name>** dialog box appears.

   Note: The **Statistics for the Configuration <config_name>** window displays all of the intermediate files that are registered for the configuration. The **Intermediate Files for the Configuration <config_name>** dialog box displays the intermediate files for which details are available.
3. Enter a target connection in one of the following ways:
   - In the **Statistics for the Configuration <config_name>** window, select a connection in the **Target** list.
   - In the **Intermediate Files for the Configuration <config_name>** dialog box, select a connection in the **Target database connection** list.

   The Data Replication Console shows a list of available intermediate files.
4. Select the intermediate file for which to view details and click **View Details**.

The **Intermediate File Information** window appears:



The upper list box lists the intermediate file records that match the selected record type and date range. When you select an intermediate file record, the **Columns** list box lists the columns for that record and any changes to the column value and size. For Update records, unchanged columns are indicated by gray shading in the **New Value** and **New Size** cells. For more information, see "Change Record Information in Intermediate Files" on page 321.

5. Click **Data** to view records in the .dat file, or click **Transactions** to view records in the corresponding .trn file.

   For more information about these records, see "Change Record Information in Intermediate Files" on page 321.

6. To filter change records, enter one or more of the following filter criteria:

   - In the **Records** field, select one or more of the **Insert**, **Update**, and **Delete** check boxes to filter records by the DML operation type.
   - In the **Other types** list, select other record types.
   - To filter records by transaction ID, enter the transaction ID in the **Filter by Xid** box and then press Enter.

     Tip: To copy a transaction ID to the **Filter by Xid** field, select a record row in the upper list box, right-click the row, and click **Filter by Xid**.
   - To filter records by date and time, select the check box that precedes the **From** and **to** fields and enter date and time values in these fields. Then click **Show**.
   - To filter records by minimum and maximum SCN values, enter SCN values in the **Filter by min SCN** and **Filter by max SCN** fields and then press Enter.

Tip: To copy an SCN value to the **Filter by min SCN** or **Filter by max SCN** field, select a record row in the upper list box, right-click the row, and then click one of the following options:

- **Filter by min SCN**. Copies the SCN value to the **Filter by min SCN** field.
- **Filter by max SCN**. Copies the SCN value to the **Filter by max SCN** field.
- **Filter by SCN**. Copies the SCN value to the **Filter by min SCN** and **Filter by max SCN** fields.

7. To open the preceding or next intermediate file, click **Back** or **Next**.
8. When you are finished, click **Close**.

Related Topics:

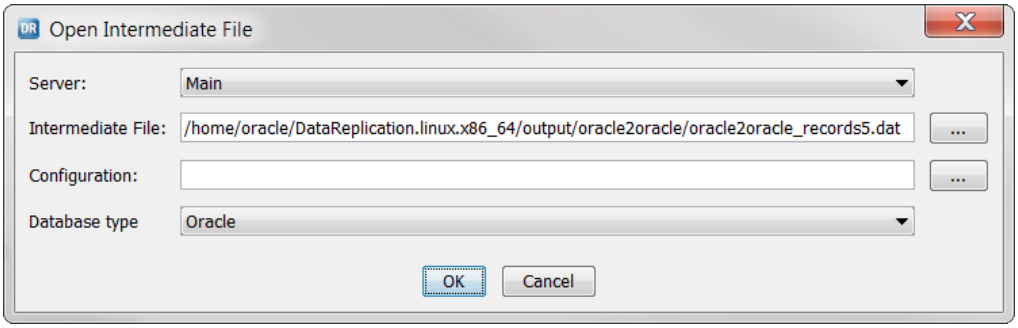
- "Viewing a Specific Intermediate File Directly" on page 319
- "Change Record Information in Intermediate Files" on page 321

Viewing a Specific Intermediate File Directly

In the Data Replication Console, you can view the content of the .dat and .trn files that comprise a specific intermediate file directly, without drilling down from a configuration. This feature is primarily for use by the Data Replication administrator and Informatica Global Customer Support.

1. Click **Tools** > **View Intermediate File**.

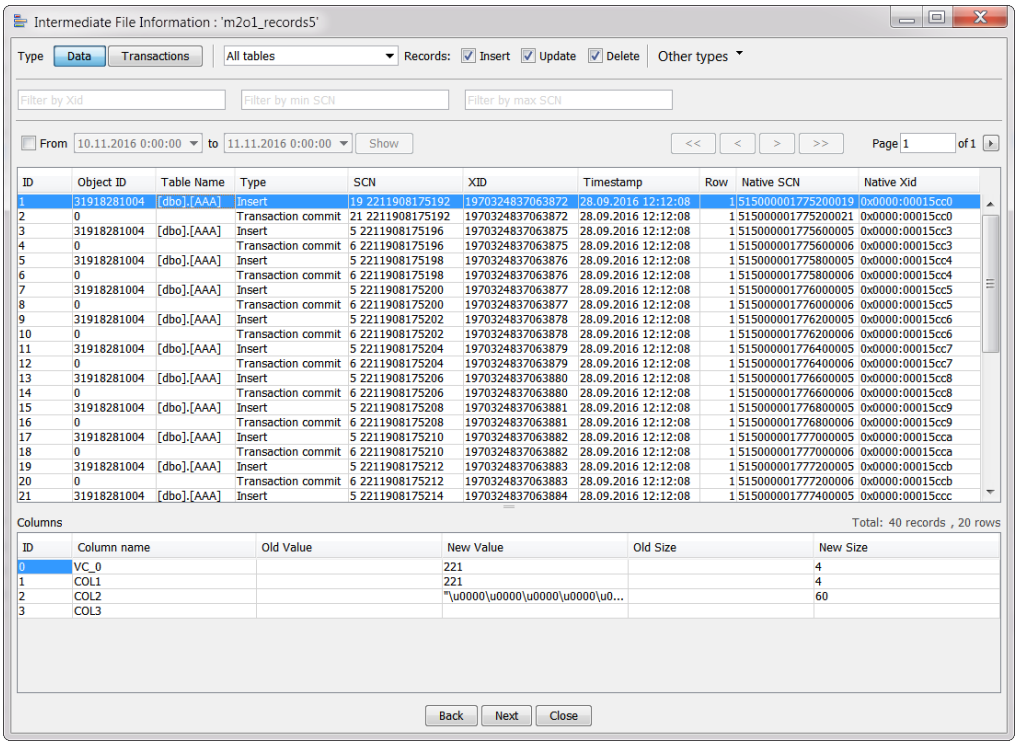
   The **Open Intermediate File** dialog box appears:



2. In the **Server** list, select the Server Manager instance that stores intermediate files.
3. In the **Intermediate File** field, enter the full path to the intermediate file that you want to view or click the **Browse** button to browse to the intermediate file.
4. Optionally, in the **Configuration** field, enter the full path to the configuration SQLite database or click the **Browse** button to browse to the configuration SQLite database.

   Important: The Data Replication Console requires a configuration SQLite database to decrypt encrypted intermediate files and to translate table and column identifiers that are used in intermediate files into table and column names. If you do not specify a configuration SQLite database for an encrypted intermediate file, the Data Replication Console cannot read the intermediate file. If you do not specify a configuration SQLite database for an intermediate file that is not encrypted, the table and column names on the **Intermediate File Information** window will contain empty values.
5. In the **Database type** list, select the source database type.
6. Click **OK**.

The **Intermediate File Information** window appears:



The upper list box lists the intermediate file records that match the selected record type and date range. When you select an intermediate file record, the **Columns** list box lists the columns for that record and any changes to the column value and size. For Update records, unchanged columns are indicated by gray shading in the **New Value** and **New Size** cells. For more information, see "Change Record Information in Intermediate Files" on page 321.

7. Click **Data** to view records in the .dat file, or click **Transactions** to view records in the corresponding .trn file.

   For more information about these records, see "Change Record Information in Intermediate Files" on page 321.

8. To filter change records, enter one or more of the following filter criteria:

   - In the **Records** field, select one or more of the **Insert**, **Update**, and **Delete** check boxes to filter records by the DML operation type.
   - In the **Other types** list, select other record types.
   - To filter records by transaction ID, enter the transaction ID in the **Filter by Xid** box and then press Enter.

     Tip: To copy a transaction ID to the **Filter by Xid** field, select a record row in the upper list box, right-click the row, and click **Filter by Xid**.
   - To filter records by date and time, select the check box that precedes the **From** and **to** fields and enter date and time values in these fields. Then click **Show**.
   - To filter records by minimum and maximum SCN values, enter SCN values in the **Filter by min SCN** and **Filter by max SCN** fields and then press Enter.

Tip: To copy an SCN value to the **Filter by min SCN** or **Filter by max SCN** field, select a record row in the upper list box, right-click the row, and then click one of the following options:

- **Filter by min SCN**. Copies the SCN value to the **Filter by min SCN** field.
- **Filter by max SCN**. Copies the SCN value to the **Filter by max SCN** field.
- **Filter by SCN**. Copies the SCN value to the **Filter by min SCN** and **Filter by max SCN** fields.

9. To open the preceding or next intermediate file, click **Back** or **Next**.
10. When you are finished, click **Close**.

Related Topics:

- "Viewing the Intermediate Files for a Configuration" on page 317
- "Change Record Information in Intermediate Files" on page 321

Change Record Information in Intermediate Files

The **Intermediate File Information: <file_name>** window in the Data Replication Console shows the content of the change records in an intermediate file.

The upper list box lists the records in the intermediate file. Typically, each record in the intermediate file represents a single SQL operation that occurred on the source. However, bulk SQL operations, such as INSERT ... SELECT, can be represented by multiple records in the intermediate file.

The following table describes the columns of information that are displayed for each record in the upper list box:

Column	Description
ID	Identifier for the record in the list.
Object ID	For .dat files, this column shows the 8-byte object ID where the first 4 bytes contain the database ID and the last 4 bytes contain the table ID. For .trn files, this column shows the offset of the record in bytes.
Table Name	Name of the table in which the operation occurred. **Note:** If you did not specify the associated configuration, this column contains empty values.
Type	Type of SQL operation that is associated with the record.
SCN	SCN value for the record.
Xid	Transaction ID.
Timestamp	Date and time when the operation occurred.
Row	Record ID in the SQL operation. For bulk SQL operations that include multiple records, this column value identifies a record in the operation. For the SQL operations that include one record, this column value is 1.
Native SCN	SCN value for the record in the source database format.
Native Xid	Transaction ID in the source database format.

The lower list box displays the before and after image data from the .dat file for the record that is selected in the upper list box.

The following table describes the columns of information in the lower list box:

Column	Description
ID	Identifier for the table row.
Column Name	Name of the column. **Note:** If you did not specify the associated configuration, this column contains empty values.
Old Value	The before image of the column.
New Value	The after image of the column. **Note:** For Update records, if a column has supplemental logging enabled and does not have an updated value, the **New Value** cell is shaded gray to indicate that there is no change to the column.
Old Size	The size of the before image.
New Size	The size of the after image. **Note:** For Update records, if a column has supplemental logging enabled and does not have an updated value, the **New Size** cell is shaded gray to indicate that there is no change to the column.

If the amount of LOB data in a source column exceeds the maximum amount that is specified in the IFViewersMaxBlobSize parameter, the **Old Value** and **New Value** column values are truncated to the specified maximum size. To view the entire LOB value, either double-click the **Old Value** or **New Value** cell, or right-click the cell and click **Show old value** or **Show new value**.

Related Topics:

- "Viewing the Intermediate Files for a Configuration" on page 317
- "Viewing a Specific Intermediate File Directly" on page 319

Task Execution Logs

For replication tasks, you can view execution logs and statistics to analyze and diagnose potential problems.

For configurations that have multiple targets, Data Replication provides a separate log for each target process of a task.

Data Replication stores the logs for the active tasks in the *DataReplication_installation*/logs directory on the system where the Server Manager for the tasks runs. When a task starts, Data Replication creates a directory that is named for the current date in the *DataReplication_installation*/logs subdirectory, if the directory does not already exist. Within this current-date directory, Data Replication creates a subdirectory for each task that runs, using the task identifier as the subdirectory name. When a task completes, the Server Manager transmits the log files to the Server Manager Main server. The Server Manager Main server stores the log files locally in the *DataReplication_installation*/logs directory, using a subdirectory hierarchy that is identical to that on the system where the task runs.

Every hour, the Server Manager scans the *DataReplication_installation*/logs subdirectories that contain log files and purges the old log files.

The Server Manager retains log files based on the following criteria:

- For tasks that run as a part of a schedule, the Server Manager uses the schedule **Keep logs, days** setting to determine the period for which to retain the log files. By default, the Server Manager stores the log files for 30 days.
- For a task that runs outside of a schedule, the Server Manager retains log files for 30 days after the task completes.
- If you delete a schedule, the Server Manager immediately purges the log files for all of the tasks in the schedule.

For convenience, you can view the log files for tasks from the **Server Manager** tab > **Schedules** view or the **Server Manager** tab > **Event Viewer** view > **Logs of Schedules** subtab in the Data Replication Console. After the Server Manager purges the log files, they are no longer available for viewing in the Data Replication Console.

For the InitialSync and Applier tasks, Data Replication periodically logs processing status information.

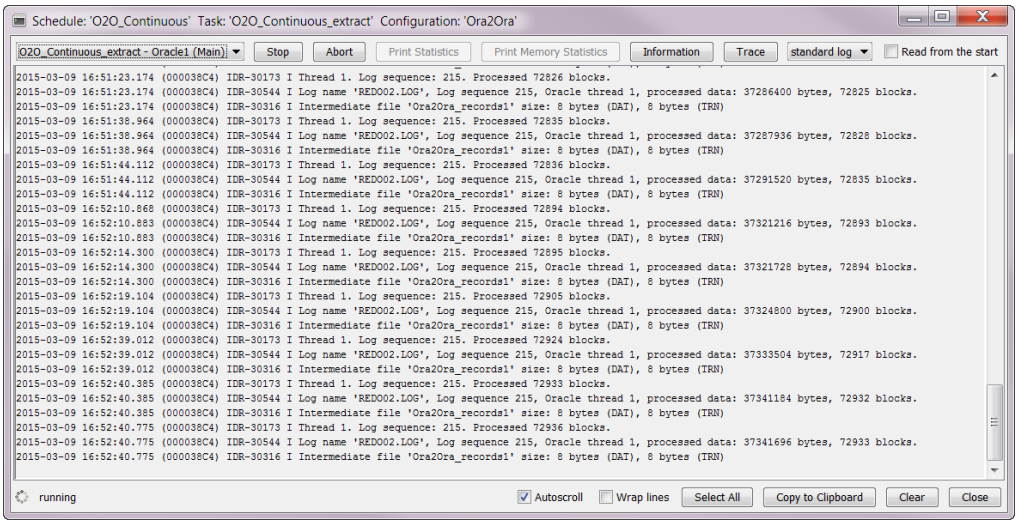
- For the InitialSync task, Data Replication updates processing statistics at 10-second intervals by default. You can use the initial.print_statistics_interval runtime parameter to change the interval. These statistics include the number of processed rows, the total number of rows to be loaded, and the estimated time for processing to complete.
- For the Applier task, Data Replication updates the number of processed rows at 5-minute intervals by default. You can use the apply.print_statistics_interval runtime parameter to change the interval. Data Replication also logs a warning message when the Applier task detects significant performance degradation.

Viewing the Current or Latest Logs for a Scheduled Task

You can view the current execution logs for a scheduled task that is running or the latest available execution logs for a scheduled task that ran previously. You can also stop or abort a running task from the log view.

1. Click the **Server Manager** tab > **Schedules** view.
2. To access the current or latest logs for a task, double-click the task row, or right-click the task row and click **View Task Log**.

   The following image shows the log view for a running task:



3. From the list of tasks in the upper left corner of the log view, select the task process for which to display the log.
4. From the list in the upper right corner of the log view, select one of the following log types:
   - **standard log**. Contains the task output.
   - **additional log**. Contains the contents of the Server Manager logs for the task execution, including command-line, error, and debug information.
5. If you selected the **standard log** type for a running Applier task, click **Print Memory Statistics** to view the following memory statistics for the Applier process:
   - Current memory usage by the Applier process.
   - Memory that is allocated to the initialized Applier process before an apply cycle starts.
   - Memory usage difference. Data Replication calculates the difference between the current memory usage and the memory allocated to the initialized Applier process.
   - Memory usage by Applier internal objects. For each Applier object, Data Replication reports memory usage in bytes and as a percentage of the *memory usage difference* that is reported for the entire Applier process.
   - The amount of data and number of records that each Applier thread has in a change data queue.
6. If you selected the **additional log** type for a running InitialSync or Applier task, click **Print Statistics** to view processing status information.

   This information includes the total number of rows that the Applier applied during the run and the average row processing time.
7. To view more information about the selected task, click **Information**.

   The **Task Information** dialog box displays the following tabs of information about the selected task:
   - The **Task Information** tab displays detailed information about the task, including the run ID, task name, start and end time, database connection, server, type of task, command syntax, directory, success or failure status, and exit code.
   - The **Log Files** tab displays the log file names and file sizes in bytes. You can also open or download the task log file.
   - The **Debug Information** tab provides options for editing the debug logging and verbosity settings for the task.

     Note: This tab is available only while the task is running.
   - The **General Information** tab displays the task name, type, Server Manager, command syntax, and the task owner.
8. To view stack trace information for a running Applier, Extractor, or InitialSync task, click **Trace**.
9. To read a large log from the beginning, select **Read from the start**.
10. To improve the legibility of the log text, select the **Autoscroll** or **Wrap lines** option.
11. To stop a running task, click **Stop**. In urgent cases where you cannot wait for a task to complete, right-click the schedule row and click **Abort**.
12. To copy the log contents to the Clipboard so that you can paste it into a file, click **Select All** and then click **Copy to Clipboard**.

    Tip: This feature is useful for sending the log contents to Informatica Global Customer Support for analysis when diagnosing a problem.

Viewing History for Schedule Instances and Tasks

The Event Viewer of the Server Manager shows a journal of events for all tasks and schedules that run under the Server Manager.

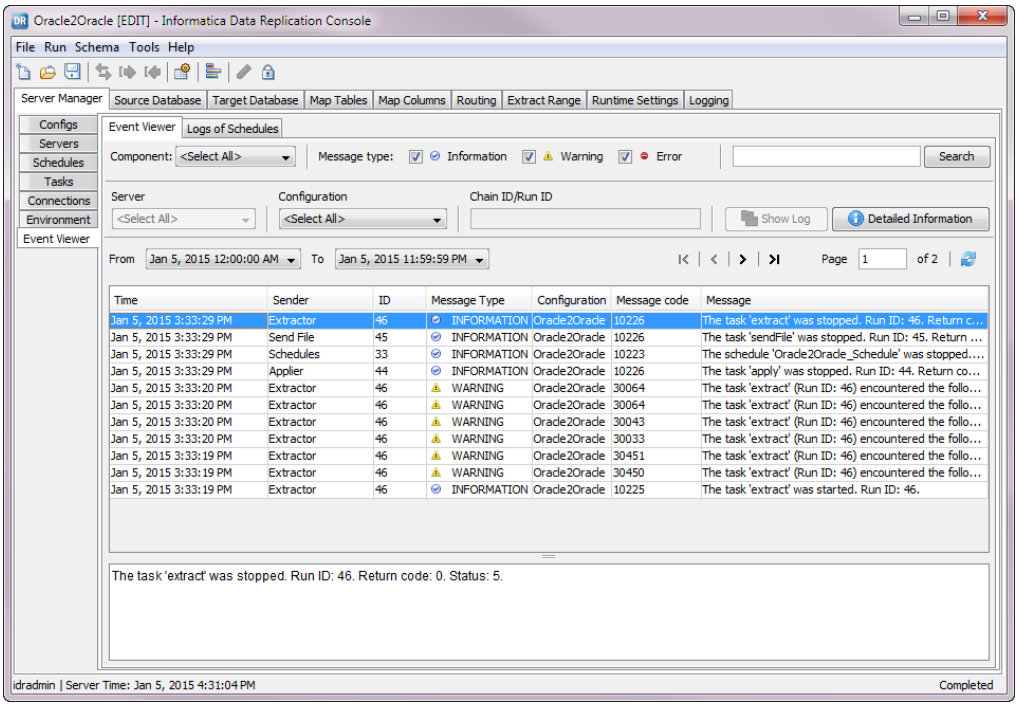
You can also view a list of schedule instances that are running or that have previously run and the tasks that are associated with these schedule instances. Drill down on a task to view its logs.

You can filter the lists of events and schedule instances based on multiple criteria.

If you are the idradmin user, you can configure the types of events to include in the journal. Click **File** > **Settings**. In the **Settings** dialog box, click the **Subscribers** tab > **Journal** subtab. Then select the event types to include in the journal and click **Save**. Event types include schedule start, schedule end, schedule error, task start, task end, task error, recovery task, task restart, task warning, Server Manager information, Server Manager warning, and Server Manager error.

1. Click the **Server Manager** tab > **Event Viewer** view.

   The following image shows an example **Event Viewer** view:



2. To filter the list of task events in the **Event Viewer** view, enter one or more of the following filter criteria:
   - In the **Component** field, select a Data Replication component.
   - In the **Message type** field, clear any of the following check boxes to filter the list based on message severity: **Information**, **Warning**, and **Error**.
   - In the text box next to the **Search** button, enter a word or phrase from a message.
   - In the **Server** list, select a Server Manager instance.
   - In the **Configuration** list, select a replication configuration.
   - In the **Chain ID/Run ID** field, enter a schedule instance ID or a task instance ID.
   - In the **From** field and the **To** field, enter a date and time to define a time period during which the events occurred.

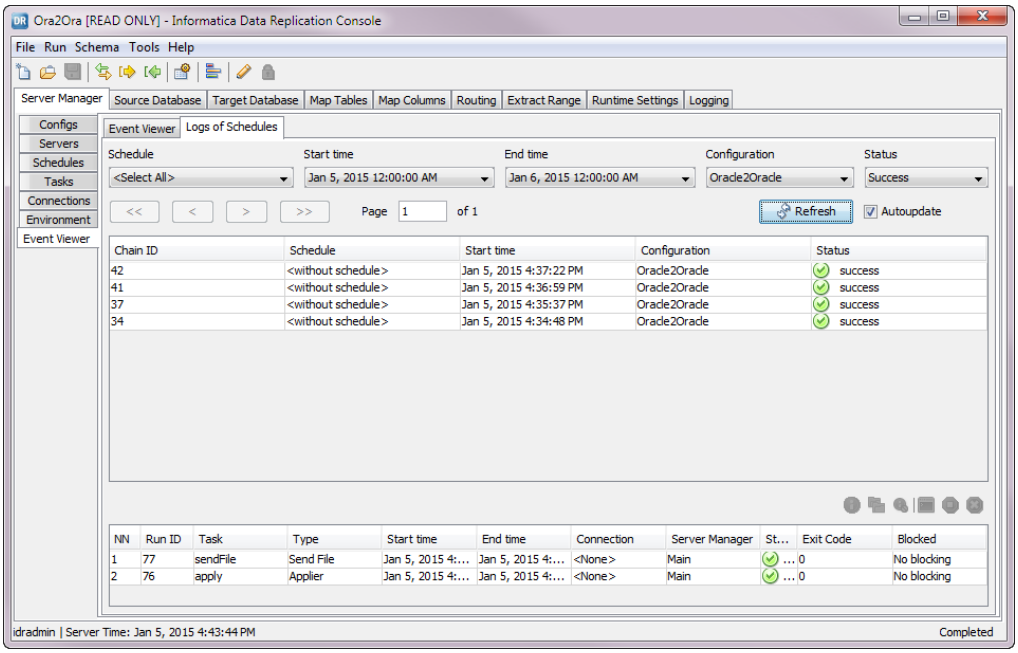
   Then click the **Refresh** icon button.

A filtered list of events appears. The list shows the event date and time, sender component, schedule instance ID (chain number) or task instance ID (run number), message type, configuration name, message code, and message about the event.

Tip: Use the |<, <, >, and >| buttons and the **Page** field to navigate through the list.

3. To view more information about an event, select the event in the list and click **Show Log** or **Detailed Information**. You can also right-click a listed event and select **Show Log** or **Detailed Information** from the shortcut menu.
4. To view the list of current and previous schedule executions within the specified time period, click the **Logs of Schedules** tab.

   The following image shows an example **Logs of Schedules** tab:



   Note: For tasks that were started manually, the **Schedule** column displays the value <without schedule>.
5. To filter the list of schedule instances, define any of the following filter criteria in the list boxes at the top:
   - In the **Schedule** list, select a schedule name.
   - In the **Start time** field and the **End time** field, enter a date and time to define the time period during which the schedule instances ran.
   - In the **Configuration** list, select the configuration name that is associated with the schedule.
   - In the **Status** list, select one of the following schedule statuses: **Running**, **Success**, or **Failure**.

   Then click **Refresh** to display the filtered list, or select the **Autoupdate** check box to automatically refresh the filtered list every 5 seconds.
6. To view task details for a schedule instance, select the schedule instance in the upper list box.

   The lower list box displays the tasks for the selected schedule. Optionally, view more information about these tasks by performing any of the following actions:
   - To view more information about a specific task, right-click the task row and select **Task Information**, or select the task row and click the **Task Information** icon button.

The **Task Information** tab of the **Task Information** dialog box appears. This tab displays detailed information about the task, including the run ID, task name, start and end times, database connection, server, type of task, command syntax, directory, success or failure status, and exit code.

- To view the logs for a specific task, right-click the task row and select **Show Log Files**, or select the task row and click the **Show Log Files** icon button.
  The the **Log Files** tab of the **Task Information** dialog box appears. From this tab, you can open or download a log file. Informatica recommends that you download log files to your local machine and then open them with an external text editor.

  Note: If you replicate data from one source to multiple targets, the **Log Files** tab in the **Task Information** dialog box lists the log files for each target.
- To view information about any processes that are blocking a task that is currently running, right-click the task row and select **Blocking Information**, or select the task row and click the **Blocking Information** icon button.
- To view the process logs for a task that is currently running, right-click the task row and select **View Process Log**, or select the task row and click the **View Process Log** icon button.
  The log view for the running task appears. For more information about this view, see "Viewing the Current or Latest Logs for a Scheduled Task" on page 323.
- To manually stop a running task, right-click the task row and select **Stop**, or select the task row and click the **Stop** icon button.
- If the situation is urgent and you cannot wait for a running task to complete processing, right-click the task row and select **Abort**, or select the task row and click the **Abort** icon button to force the task to end.

Task Exit Codes

In the **Server Manager** tab > **Logs** view, you can view the exit codes for replication tasks that completed processing successfully or ended abnormally.

If a replication task ends abnormally, use the exit code to diagnose the problem.

The following table describes the exit codes of replication tasks:

Exit Code	Description
0	The replication task completed processing successfully.
30	The replication task ended abnormally because of an SQLite repository error.
31	The replication task could not connect to the source database.
32	The replication task could not connect to the target database.
33	InitialSync failed to use a Flashback query to unload data from the Oracle source.
34	The replication task ended abnormally because of a database error.
35	The replication task could not find the XML configuration file.
36	The replication task could not find the XSD schema file.
37	The replication task could not parse the XML configuration file because the file is corrupted.

Exit Code	Description
38	Internal error. The replication task could not process the transform.xsl file. Contact Informatica Global Customer Support.
39	The replication task did not process change data from a column because the column datatype is not supported. Edit the replication configuration to unmap the column with the unsupported datatype.
40	The replication task could not build the ODBC connection string.
41	InitialSync did not resynchronize tables because no tables were selected for resynchronization.
42	The Extractor task could not process WHERE conditions for source columns.
43	The replication task could not parse an intermediate file. The intermediate file might be corrupted.
44	Internal error. A memory allocation error occurred. Contact Informatica Global Customer Support.
45	An I/O operation failed.
47	The replication task was started with command line arguments that were not valid. For more information about the command line arguments that the replication task supports, see Appendix D, "Command Line Parameters for Data Replication Components" on page 438.
49	An operating system error occurred.
50	The replication task ended abnormally because of an error in the SQLite configuration repository.
51	Internal error. Cannot save the XML configuration file. Contact Informatica Global Customer Support.
52	The replication task expected data with a different endianness.
53	The Applier did not apply change data to the target tables because these tables were not initially synchronized with the source tables. Run InitialSync before you start change data replication the first time.
54	A replication task ended abnormally because of insufficient disk space on the Data Replication system. Ensure that the system has enough free disk space and then restart the replication task.
55	InitialSync ended because the db2.initial_lsn runtime parameter does not specify an initial LSN value. For DB2 sources, unless you override the default LSN value, you must run the Extractor before InitialSync to record an initial LSN in the db2.initial_lsn parameter. The first time you run the Extractor after InitialSync completes, the Extractor begins extracting changes from this LSN.
56	The database user that Data Replication uses to connect to the source or target database has insufficient privileges. Grant the required privileges to the database user. For more information, see Chapter 3, "Sources - Preparation and Replication Considerations" on page 60.
57	InitialSync cannot synchronize source tables with target tables that are not empty. Truncate the mapped target tables and run InitialSync again.

Server Manager Logs

The Server Manager writes error and debug information to log files on the local file system. You can use these log files to diagnose Server Manager problems that might occur.

When you start the Main Server Manager or a subserver, it creates a log file that has a file name that consists of the current timestamp followed by the log suffix. The Server Manager stores log files for each execution of a Server Manager server or subserver in the *DataReplication_installation*/logs/SM subdirectory. The Server Manager never purges the log files in this subdirectory.

Viewing Server Manager Logs

After you connect to the Main Server Manager server, you can view the full execution logs for the Main Server Manager and any subserver from the Data Replication Console.
The Server Manager stores all of its log files in the *DataReplication_installation*/logs/SM subdirectory.

1. Click the **Server Manager** tab > **Servers** view.
2. Select the server for which to view logs.
3. Click the **View Logs** button on the toolbar, or right-click the server row and click **View Logs**.

   The **Log for the Server <server_name>** window opens and displays the log content for the selected Server Manager instance.

   - To view memory statistics for Server Manager processes, click **Print Memory Statistics**.
   - To view the stack trace for the log, click **Trace**.
   - To look for a particular point in a log, clear the **Autoscroll** box.
   - If some lines are too long to display entirely in the window area, select **Wrap lines**.
   - To copy the contents to a file, use the **Select All** and **Copy to Clipboard** buttons.
   - To delete the log information from the **Log for the Server <server_name>** window, click **Clear**. Deleting this information from the Console does not affect the contents of the log file in the *DataReplication_installation*/logs/SM subdirectory. The **Clear** button provides a way for you to view less log file information and conserve Console memory.

Note: The idradmin user can set the logging level and suffix for the log files. Click the **Server Manager** tab > **Servers** view. Then select the server row and click **Properties**. On the **Logs** subtab, enter the suffix and logging level and click **Save**.

User Notifications

After you connect to a Server Manager instance, you can configure email and SNMP user notifications of Data Replication events by subscriber and event type. Optionally, you can attach message log files to the notifications.

Configuring Email Notifications

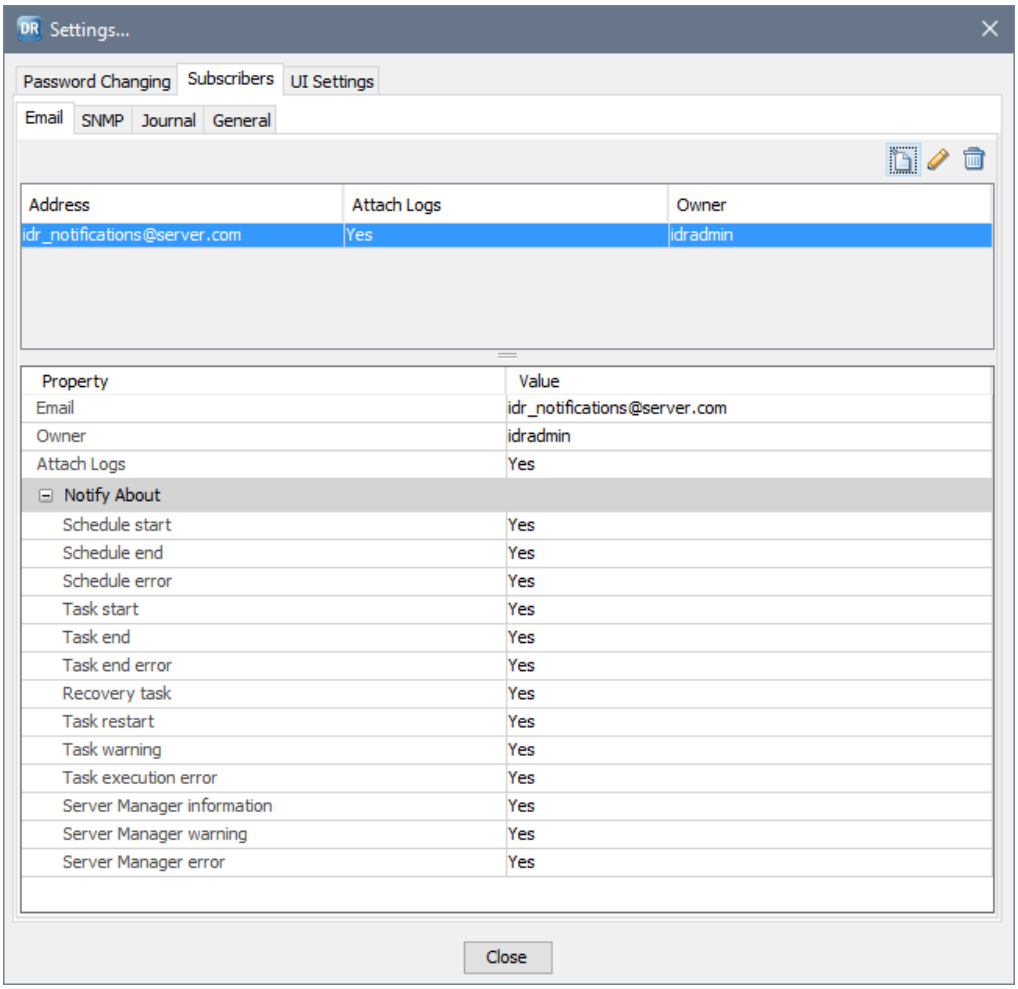
You can configure the Server Manager to send email notifications with attached logs to subscribers for certain types of events.

1. Click **File** > **Settings**.

   The **Settings** dialog box appears.

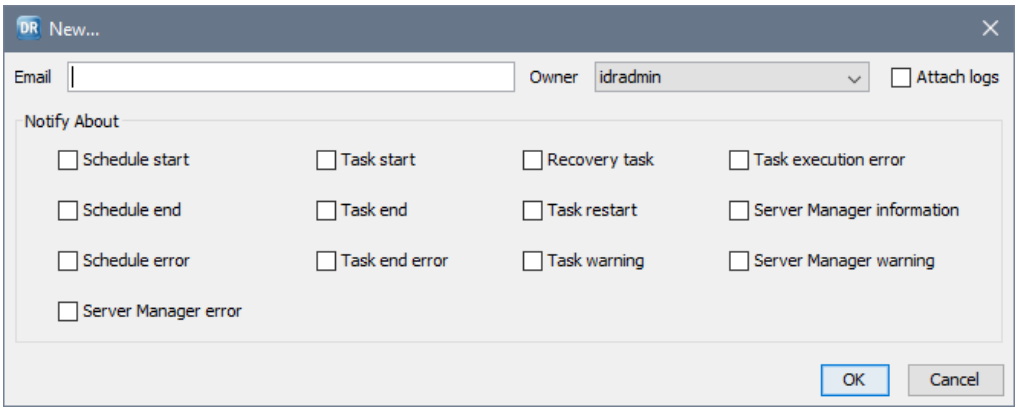
2. Click the **Subscribers** tab > **Email** subtab.

   The following image shows the **Email** subtab:



3. To add a subscriber, click the **New** icon button.

The **New** dialog box appears.



4. In the **Email** field, enter the email address of a subscriber.
5. In the **Owner** list, select an owner for the subscriber.
6. Under **Notify About**, select the events for which to notify the subscriber.
7. To attach message log files to user notifications, select **Attach logs**.
8. Click **OK**.

   The subscriber is listed on the **Email** subtab.

   Tip: To edit an email subscriber for which you are designated as the owner, click the **Edit** icon button. To delete an email subscriber for which you are designated as the owner, click the **Delete** icon button.
9. Repeat Steps 3-8 to add more subscribers.

To enable user notifications by email, you must also configure the Server Manager email server settings. See "Configuring the Server Manager Email Server and SNMP Settings" on page 333.

Note: By default, the Server Manager waits 10 minutes after sending an email notification to users about a specific event type before sending another email notification for the same event type. You can use the Server Manager SendMailTimeOut advanced property to change the amount of time that the Server Manager waits before sending another email notification.

Configuring SNMP Notifications

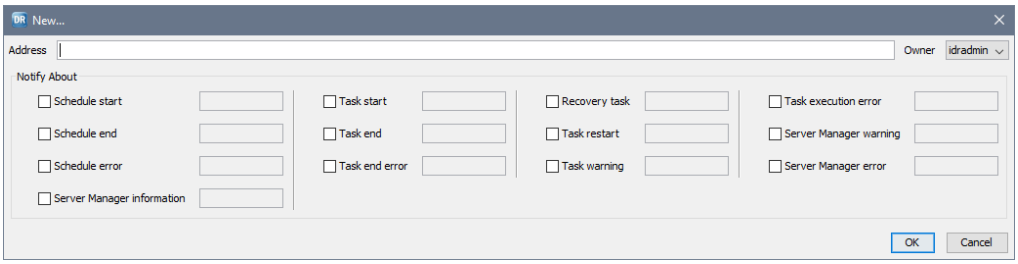
You can configure the Server Manager to send SNMP notifications to an SNMP manager for certain types of events.

Before you begin, you must configure the SNMP manager that will receive SNMP traps from the Server Manager.

1. Click **File** > **Settings**.

   The **Settings** dialog box appears.
2. Click the **Subscribers** tab > **SNMP** subtab.
3. Click **New** icon button to add an SNMP manager.

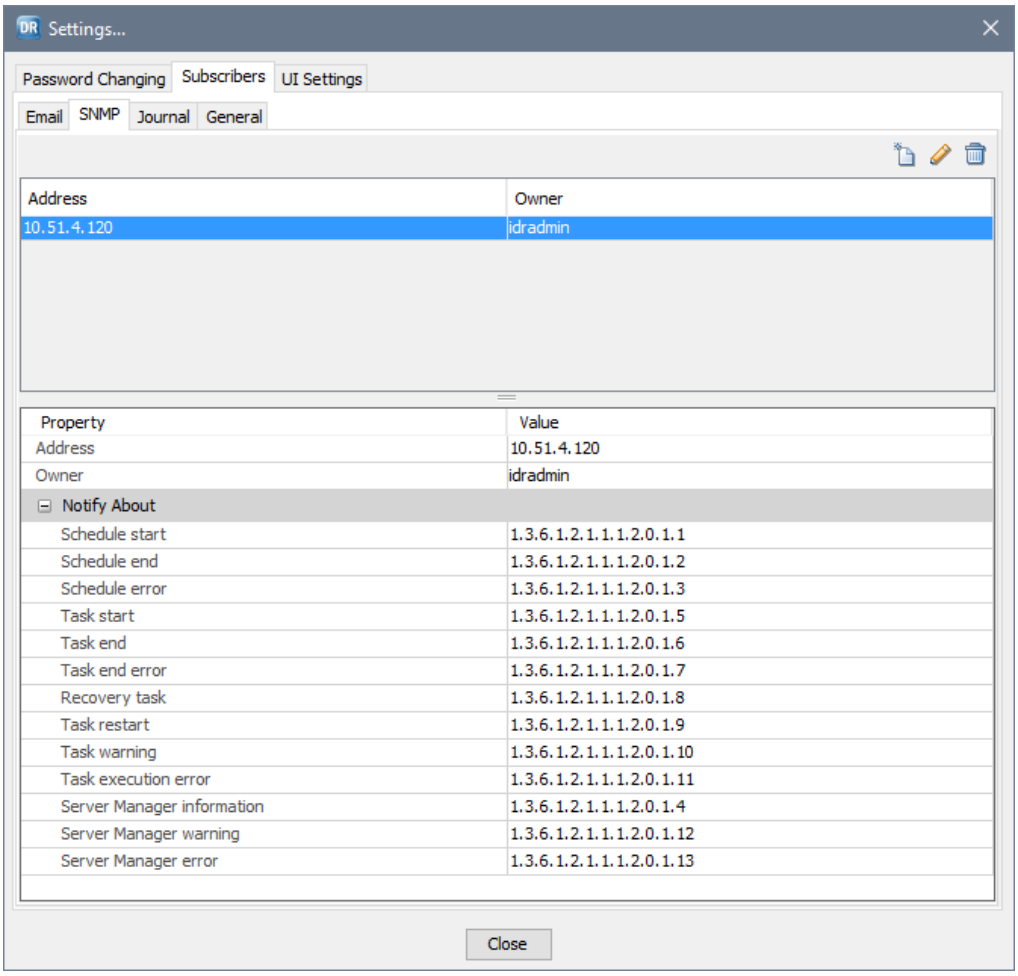
The **New** dialog box appears.



4. In the **Address** field, enter the IP address of the SNMP manager that you configured to receive SNMP traps from this Server Manager.
5. In the **Owner** list, select an owner for the subscriber entry for the SNMP manager.
6. Under **Notify About**, select the events for which to notify the SNMP manager and provide the object identifier (OID) for each selected event.

   Important: Object IDs for events must start with the Enterprise Object ID that you specified in the **SNMP community OID** field on the **General** tab.
7. Click **OK**.

   The following image shows the **SNMP** subtab, which lists the new subscriber entry:



Tip: To edit an SNMP subscriber for which you are the owner, click **Edit** icon button. To delete an SNMP subscriber for which you are the owner, click the **Delete** icon button.

To enable user notifications by SNMP, you must also configure the Server Manager SNMP settings. See "Configuring the Server Manager Email Server and SNMP Settings" on page 333.

Configuring User Journals of Events

The idradmin user can configure the types of events that users can see on the **Server Manager** tab > **Logs** view > **Journal** subtab.

1. On the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user.
2. Click **File** > **Settings**.
3. In the **Settings** dialog box, click the **Subscribers** tab > **Journal** subtab.
4. Select the types of events to journal.
5. Click **Save**.

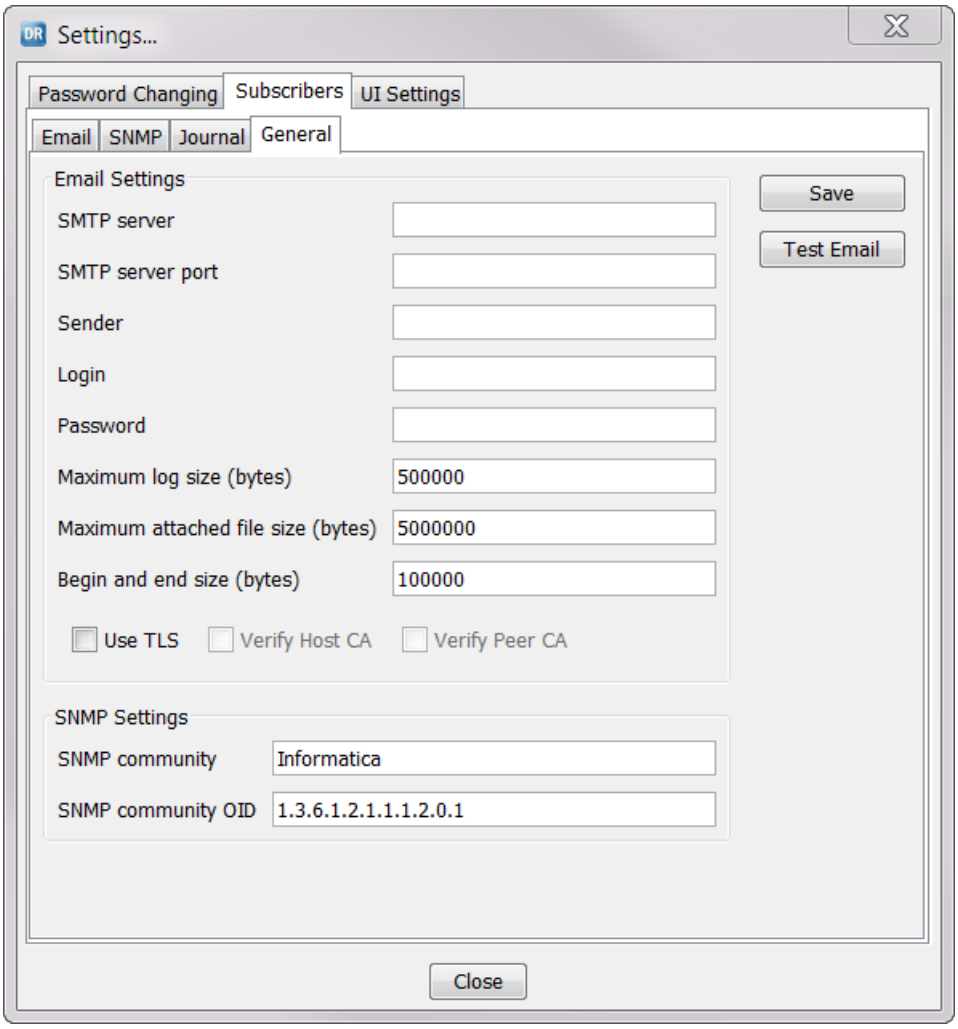
Configuring the Server Manager Email Server and SNMP Settings

To enable user notifications by email and SNMP, configure Server Manager email and SNMP settings. Only the idradmin user can configure the Server Manager settings.

1. On the **Server Manager** tab > **Configs** view, connect to the Main server as the idradmin user.
2. Click **File** > **Settings**.

   The **Settings** dialog box appears.
3. Click the **Subscribers** tab > **General** subtab.

The following image shows the **General** subtab:



4. Enter Server Manager email settings.

   The following table describes these settings:

Field	Description
SMTP Server	The SMTP Server address for the Server Manager.
SMTP Server port	The SMTP Server port number that the Server Manager uses to send email notifications. Valid values are integers from 1 through 65535.
Sender	The email address that the Server Manager uses to send email notifications.
Login	The user login for the SMTP Server.
Password	The password for the SMTP Server.

Field	Description
Maximum log size (bytes)	The maximum size of the log files, in bytes, that the Server Manager can paste into the email body.
Maximum attached file size (bytes)	The maximum size of the log files, in bytes, that the Server Manager can attach to an email.
Begin and end size (bytes)	The size of the start and end blocks in log files, in bytes, that the Server Manager pastes into the email body when the log file is too large to be attached.
Use TLS	Select this option if the SMTP Server uses HTTPS connections.
Verify Host CA	Select this option to verify that the SSL certificate is associated with the host name of the SMTP Server.
Verify Peer CA	Select this option to verify the authenticity of the SSL certificate.
SNMP community	An override for the default SNMP community string. Data Replication sends SNMP traps to the Informatica community by default. If you keep the default value, ensure that the SNMP manager that you configure to receive SNMP notifications accepts SNMP traps from this community. **Note:** Most SNMP managers use the community name of "public" by default.
SNMP community OID	An override for the SNMP community object identifier (OID). An enterprise OID override consists of a series of digits that are separated by periods. **Important:** Contact your system administrator to ensure that the SNMP community OID value does not conflict with enterprise OIDs of other SNMP agents in your network.

5. To send a test email, click **Test Email**.
6. Click **Save**.

Skipped Transaction Records

By default, when the Extractor cannot parse a record in the source transaction logs, the Extractor logs an error and stops processing to avoid data integrity issues. You can use the extract.stop_on_parsing_error runtime parameter to have the Extractor skip the damaged log record and continue processing, by default.

For Oracle sources, when a parsing error occurs, the Extractor also creates an SQLite database file that contains the unparsed transaction record and additional troubleshooting information in the *DataReplication_installation*/dump subdirectory. If you set the extract.stop_on_parsing_error parameter to 0 to skip damaged log records, you must periodically check if the *DataReplication_installation*/dump subdirectory is empty or contains files. If the dump subdirectory contains files, the Extractor skipped some records and the target database might have data integrity issues.

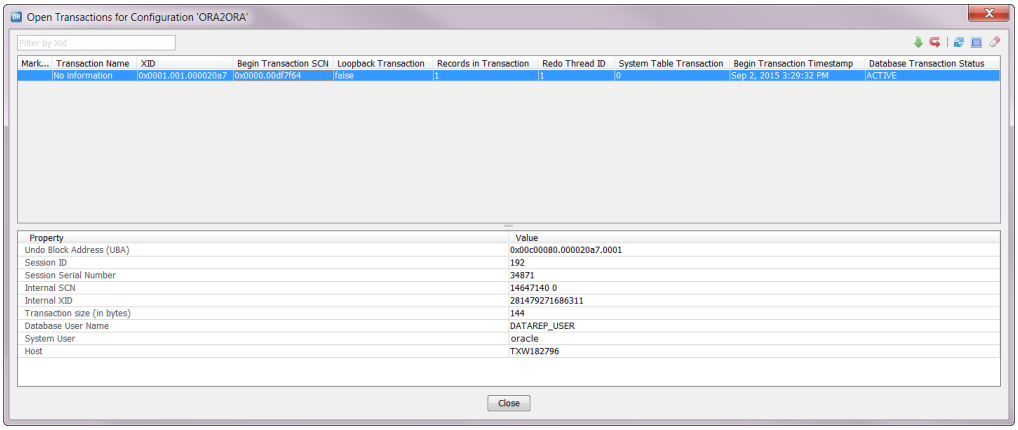
For source databases other than Oracle, the Extractor logs errors but does not create SQLite database files that contain unparsed transaction records in the *DataReplication_installation*/dump subdirectory.

Managing Open Transactions

In the Data Replication Console, you can view a list of open transactions for tables in the source database if you are an idradmin user or the owner of the selected configuration. You can commit or roll back corresponding transactions in the intermediate files to persist changes in the target database, if necessary.

1. On the **Server Manager** tab > **Configs** view, select a configuration for which to view open transactions.
2. To view the list of transactions that were open in the source database at the time of the last Extractor checkpoint, perform one of the following actions:
   - Click the **Show Transactions** icon button on the Replication Configurations toolbar.
   - Right-click the configuration row and click **Show Transactions**.

   The **Open Transactions for Configuration <configuration_name>** dialog box appears.



The list of open transactions appears at the top. The transactions for unmapped tables that do not have a **Begin Transaction Timestamp** value appear first, and then the other transactions are sorted in chronological order from earliest to most recent **Begin Transaction Timestamp** value. You can filter the list by entering an XID in the text box in the top left corner. When you select a transaction in the list, the properties box beneath the list displays additional properties for the selected transaction.

Note: For DB2 sources, the list might include transactions that DB2 processes such as DB2HMON and DB2ACD open for maintenance tasks. These transactions are not related to data replication.

The following table describes the properties that are displayed for each open transaction in the transactions list and in the properties box:

Property	Description
Mark for Commit or Rollback	Indicates if you marked the transaction for commit or rollback processing on the target.
Transaction Name	The name of the open transaction.

Property	Description
XID	The transaction ID of the open transaction. **Note:** For Oracle sources, the **Open Transactions for Configuration <configuration_name>** dialog box might list multiple open transactions with the same XID value if Oracle reuses transaction IDs. When listing these transactions, the Data Replication Console first lists the transactions for unmapped tables that do not have a **Begin Transaction Timestamp** and then lists the other transactions in chronological order, beginning with the transaction that has the earliest **Begin Transaction Timestamp**.
Begin Transaction LSN or Begin Transaction SCN (for Oracle sources)	The SCN or LSN value for the BEGIN TRANSACTION operation.
Loopback Transaction	Indicates whether the Extractor skips the transaction based on the transaction name. The Extractor skips transactions if their names are specified in the SKIP_TX command line parameter. If the SKIP_DEFAULT_TX parameter is set to Y, the Extractor also skips transactions that have the default name of DbSyncTransaction. Valid values are: - **true**. The Extractor skips the transaction. - **false**. The Extractor does not skip the transaction.
Records in Transaction	The number of records in the transaction that are available in intermediate files.
Database ID or Redo Thread ID (for Oracle sources)	For Oracle sources, the unique identifier of a redo thread. For other sources, the unique identifier of a database.
DDL Transaction or System Table Transaction (for Oracle sources)	Indicates whether the transaction affects database system tables. A transaction that affects database system tables usually includes DDL operations. The database system tables that Data Replication monitors are specified in the following parameters in the *DataReplication_installation*/uiconf/default.cfg file: - db2_fixed_source_tables_names - mssql_fixed_source_tables_names - oracle_fixed_source_tables_names Valid values are: - **0**. The transaction does not affect system tables. - **1**. The transaction affects system tables. **Note:** If the default.cfg file does not exist, you can create the file in the uiconf directory and then add parameters in the file. For more information, see "Default.cfg File" on page 391.
Begin Transaction Timestamp	The date and time of the first redo record for the mapped table. This value displays only if the transaction includes changes to a mapped table in the configuration. If the transaction includes changes to an unmapped source table, the **Begin Transaction Timestamp** property does not contain a value.
Database Transaction Status	The status of the transaction that is reported by the source database. If the source database does not contain information about the transaction, Data Replication sets the transaction status to NOT ACTIVE.
Undo Block Address (UBA)	For Oracle sources, specifies the Undo Block Address.
Session ID	For Oracle sources, specifies the identifier of the session that is associated with the open transaction.

Property	Description
Session Serial Number	For Oracle sources, specifies the serial number of the session that is associated with the open transaction.
Internal LSN or Internal SCN (for Oracle sources)	The SCN or LSN value for the BEGIN TRANSACTION operation in the database format.
Internal XID	The transaction ID in the database format.
Transaction size	The amount of data, in bytes, in the transaction records in the intermediate files.
Database User Name	The name of the database user who opened the transaction.
System User	The name of the system user who opened the transaction.
Host	The IP address or host name of the system with the database client that opened the transaction.
Application ID	For DB2 sources, the ID of the database client that opened the transaction.

Tip: To get the latest information for the list of open transactions, click the **Refresh** icon button.

3. If a transaction is causing a problem that you need to circumvent, you can commit or rollback the transaction in one of the following ways:
   - To mark an open transaction for commit processing on the target, select the transaction and click the **Commit** icon button. The Extractor adds a commit record for the selected transaction at the end of the next Extractor microcycle. The Applier commits this transaction on the target during the next apply cycle.
   - To mark an open transaction for rollback processing on the target, select the transaction and click the **Rollback** icon button. The Extractor adds a rollback record for the selected transaction at the end of the next Extractor microcycle. The Applier rolls back this transaction on the target during the next apply cycle.

   Caution: Committing or rolling back an open transaction in the Data Replication Console might result in data inconsistencies. For DB2 for Linux, UNIX, and Windows sources, change records that occur after the commit or rollback point in the source transaction are extracted and applied as part of the same transaction. However, for Oracle and Microsoft SQL Server sources, change records that occur after the commit or rollback point in the source transaction are not extracted and applied.

Related Topics:

- "Filtering Table Rows for Selective Replication to Different Targets" on page 301
- "Updating Source and Target Metadata for a Configuration in Non-interactive Mode" on page 453

CHAPTER 15

Managing Replication Configurations

This chapter includes the following topics:

Configuration Management Tasks

After you create replication configurations, you can work with them in the following ways:

- Edit a configuration.
- Deploy a configuration locally to generate additional configurations for a replication that has multiple sources.
- Deploy a configuration remotely to transfer the configuration or configuration changes to another Server Manager Main server on another system.
- Import a configuration that you manually created into the SQLite database.
- Export a configuration in XML format for Informatica Fast Clone use.
- View earlier configuration revisions.

The idradmin user can perform any configuration management tasks. Other users can edit and deploy only the configurations for which they are designated as the owner.

Switching to Read or Edit Mode for a Replication Configuration

In the Data Replication Console, you can work with replication configurations in Edit or Read mode. Switch to Edit mode to edit the configuration, and switch to Read mode to run the replication job.

To switch modes, perform one of the following actions:

- To switch to Read mode, on the **Server Manager** tab > **Configs** view, double-click a configuration file in the list to open it and then click the **Switch to Read Mode** icon button on the main toolbar. If the toolbar button is not available, verify that the configuration is in Edit mode.
- To switch to Edit mode, double-click a configuration file in the list to open it and then click the **Switch to Edit Mode** icon button on the main toolbar.
  The Data Replication Console prompts you to stop running replication tasks. Click **Yes** to stop all active replication tasks. Click **No** to edit the configuration while the tasks are running.

  If you click **No** and a DDL change occurs on the source, you cannot save the edited configuration. This issue occurs because when the Extractor captures the DDL change, it updates the configuration and increases the revision number.

Editing a Replication Configuration

In the Data Replication Console, you can open a configuration for which you are the owner and switch to Edit mode to change configuration settings.

Note: Alternatively, you can use the Replication Configuration CLI to update an existing configuration from the command line.

1. In the **Server Manager** tab > **Configs** view, double-click the configuration to open it.

   Note: If you cannot connect to the source or target database, a confirmation prompt asks if you want to open the configuration without connecting to the source and target database. Click **Yes**.
2. Ensure that no DDL changes occur on the mapped source tables from the moment the Extractor processed the last redo record until you save the edited replication configuration. Informatica recommends that you perform the following substeps to minimize the time period during which the DDL operations must not occur:
   a. Ensure that the Extractor completed processing the transaction logs.
   b. Ensure that the Applier applied change data from all of the intermediate files.

      Tip: To view the list of intermediate files and their processing statuses, on the **Server Manager** tab > **Configs** view, select a configuration for which to view intermediate files. Then click the **Intermediate Files** icon button on the Replication Configurations toolbar.
   c. Stop the replication tasks and schedules for the configuration that you want to edit.
3. To switch to Edit mode, click the **Switch to Edit Mode** icon button on the main toolbar.
4. Enter changes on the available tabs, as needed.

   If you are editing a configuration without a connection to the source or target database, you can edit only the following configuration settings: connection details, DDL and DML capture settings on the **Map Columns** tab, Tcl or SQL expressions, fields on the **Extract Range** tab except for the default path for an Oracle ASM connection, parameters on the **Runtime Settings** tab, and fields on the **Logging** tab.

5. Save the configuration to the Server Manager Main server in one of the following ways:
   - To save the configuration under its current name, click **File** > **Save** on the menu bar or click the **Save** icon button on the main toolbar.
   - To save the configuration under another name, click **File** > **Save As**. Then enter a new configuration name.

Changing the Server Manager Associated with a Replication Configuration

You can change the source or target Server Manager for a configuration for which you are the owner.

1. On the **Server Manager** tab > **Configs** view, select the configuration.
2. Right-click the configuration row and click **Properties**, or click the **Properties** icon button on the Replication Configurations toolbar.

   The **Editing** dialog box appears.
3. Click the **Change Server Manager** icon button for the source Server Manager or target Server Manager.
4. In the **Server Manager Settings for the Configuration 'config_name'** wizard, select a connection in the **Connection** list and select a server in the **Server** list. Then click **Continue**.

   Important: You can select a connection only if you change a target Server Manager for configurations that have multiple targets. The connection is used to identify a primary or secondary target for which you want to change the Server Manager.
5. If the wizard displays a list of intermediate files to transmit, click **Move**.

   Data Replication pings the new server. If the ping fails, a message indicates that the server is not available. If the ping succeeds, the wizard displays a final report that shows the status of each change operation.
6. Resolve any problems that the report identifies. Then click **Finish**.
7. In the **Runtime environment** list, select an environment variables list.

   This list includes the environment variables lists that you defined in the **Server Manager** tab > **Environment** view.
8. Click **Save**.

Clearing User Replication Settings

By default, Data Replication saves the source and target connection information, Server Manager connection information, and executable file locations and loads this information the next time you start an Informatica Data Replication Console session. You can clear this information to define a new configuration.

1. On the menu bar, click **File** > **Clear User Forms**.
2. Restart the Data Replication Console for the change to take effect.

Managing Database Supplemental Logging

For DB2 for Linux, UNIX, and Windows, Microsoft SQL Server, and Oracle sources, you can manage the database supplemental logging settings from the **Map Tables** tab in the Data Replication Console.

For DB2 for Linux, UNIX, and Windows sources, you can view or edit the DATA CAPTURE settings for source tables, even if you did not confirm that you want to manage the DATA CAPTURE settings when you saved the configuration.

For Microsoft SQL Server sources, you can use the Data Replication Console to enable or disable Change Data Capture for source tables.

For Oracle sources, you can use the Data Replication Console to add or remove the source table columns that are in supplemental log groups. You can also add a supplemental log group if you did not select the option to allow the Console to generate supplemental log groups when you save a configuration.

For DB2, Microsoft SQL Server, and Oracle source databases, you can generate a script file that includes the SQL statements for these settings. You can give this script file to your database administrator if you do not have the authority to edit database supplemental logging settings.

Related Topics:

- "Saving Replication Configurations to the Main Server Manager" on page 267
- "Specifying General Runtime Settings" on page 253

Managing DB2 DATA CAPTURE Settings

For DB2 for Linux, UNIX, and Windows sources, you can use the Data Replication Console to manage the DB2 DATA CAPTURE and DATA CAPTURE CHANGES settings for source tables. To enable supplemental logging, DATA CAPTURE CHANGES must be set for the source tables.

To change DATA CAPTURE settings in the Data Replication Console, the configuration must be open in Edit mode. To only view DATA CAPTURE settings, without making changes, the configuration can be open in Read mode.

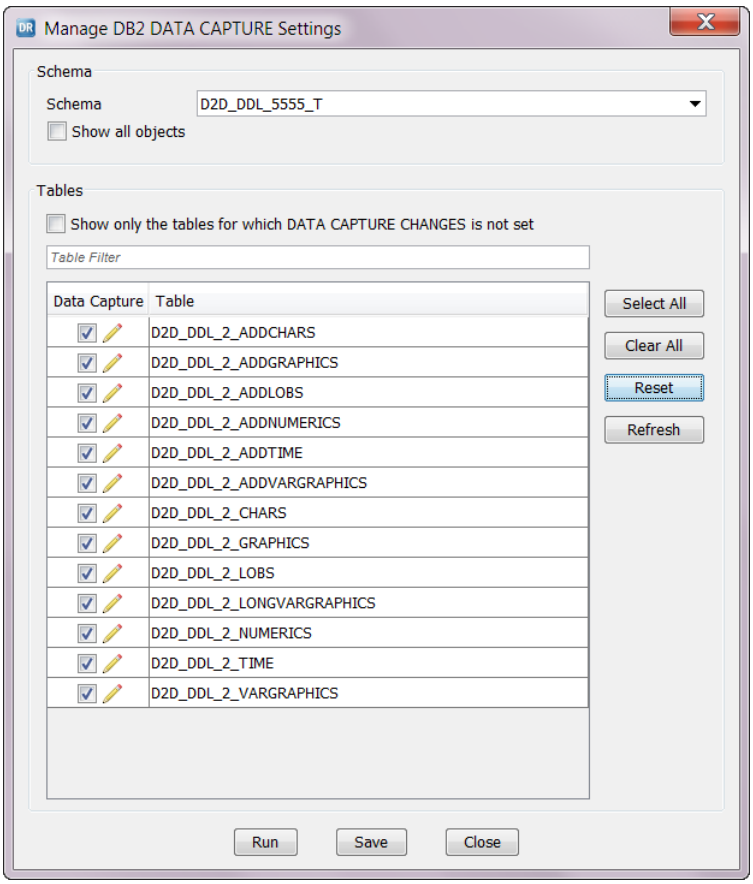
1. On the **Map Tables** tab, click the **Manage Database Supplemental Logging** icon button above the source table *Filter* list.

   The **Manage DB2 DATA CAPTURE Settings** dialog box appears.

2. In the **Schema** list, select the source schema for the tables for which you want to view or edit DATA CAPTURE settings.

   Note: The **Schema** list displays only the schemas for which one or more source tables are mapped. By default, the source schema that is currently selected on the **Map Tables** tab is displayed, if it has at least one mapped table.

A list of the mapped source tables in the specified schema appears in the **Tables** box. The following image shows the list of all mapped source tables for the D2D_DDL_5555_T schema:



Tip: To display all source tables and schemas, including unmapped tables and schemas, select **Show all objects**. To display only the tables for which DATA CAPTURE CHANGES is not set, select **Show only the tables for which DATA CAPTURE CHANGES is not set**. To filter the list of source tables, enter the first few letters of a table name in the *Table Filter* field. For case-sensitive filtering, enclose the filter value in double quotation marks.

Important: If the source schema contains thousands of tables or the Data Replication Console runs in a local area network that is remote to the database server network, selecting **Show all objects** might significantly increase metadata caching time.

3. In the **Data Capture** column, select the check box for each source table for which you want to set DATA CAPTURE CHANGES to enable supplemental logging, or clear the check box for any table for which you want to set DATA CAPTURE NONE to disable supplemental logging.

   Tip: To select the **Data Capture** check box for all of the source tables, click **Select All**. To clear the **Data Capture** check box for all of the source tables, click **Clear All**. To revert to the **Data Capture** settings that appeared when you first opened the **Manage DB2 DATA CAPTURE Settings** dialog box, click **Reset**. To refresh the Data Capture settings to get the latest changes made in the source database or from the Server Manager CLI, click **Refresh**.

   Data Replication generates SQL statements for each table for which you edited the **Data Capture** settings. Data Replication uses the following syntax for the statements:

   ```
   ALTER TABLE "schema_name"."table_name" DATA CAPTURE CHANGES INCLUDE LONGVAR COLUMNS;
   ALTER TABLE "schema_name"."table_name" DATA CAPTURE NONE;
   ```

The DATA CAPTURE CHANGES statement enables supplemental logging for the table that you selected, and the DATA CAPTURE NONE statement disables supplemental logging for the table that you selected.

4. To save the SQL statements to a script file, perform the following substeps:
   a. Click **Save**.

      The **Save** dialog box appears.

   b. Enter a file name and path and click **Save**.

      A message confirms that the script file was saved successfully.

   c. Click **OK**.

   You can use the saved script file to run the SQL statements later. If you do not have the authority to run the SQL statements or manage DATA CAPTURE settings, give the script file to a database administrator who has the authority to run the statements.
5. To run the SQL statements before exiting the dialog box, click **Run**.
6. Click **Close**.

If you set DATA CAPTURE CHANGES to enable supplemental logging for any unmapped source tables, map these tables on the **Map Tables** tab to include them in the replication configuration.

Managing Microsoft SQL Server Change Data Capture Settings

For Microsoft SQL Server sources, you can use the Data Replication Console to manage SQL Server Change Data Capture settings for source tables in a configuration. You must enable Change Data Capture to implement supplemental logging.

Important: You must run the Server Manager under a Windows administrative user.

If you plan to run the Server Manager on a computer that is remote from the SQL Server source, ensure that the **remote admin connections** option of the SQL Server sp_configure stored procedure is set to 1. This setting enables the remote Server Manager to use a dedicated administrator connection (DAC) to the SQL Server source.

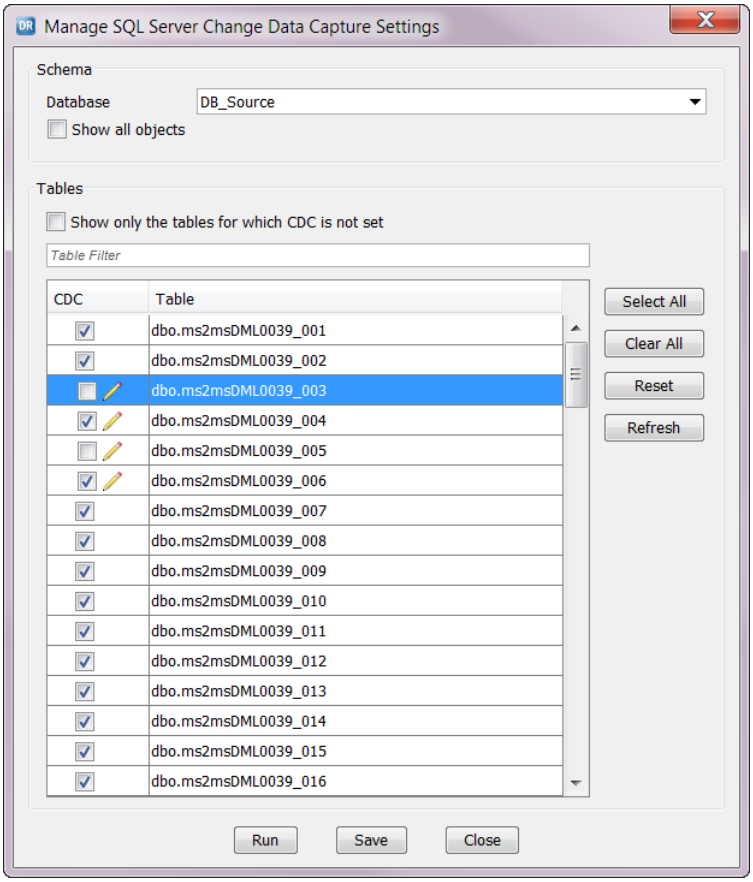
To change SQL Server Change Data Capture settings in the Data Replication Console, the configuration must be open in Edit mode. To only view Change Data Capture settings, without making changes, the configuration can be open in Read mode.

1. On the **Map Tables** tab, click the **Manage Database Supplemental Logging** icon button above the source table *Filter* list.

   The **Manage SQL Server Change Data Capture Settings** dialog box appears.
2. In the **Database** list, select the name of the SQL Server source database that contains the tables for which you want to view or edit Change Data Capture settings.

   Note: The **Database** list displays only the databases for which one or more source tables are mapped. By default, the source database that is currently selected in the **Source Schema** list on the **Map Tables** tab is displayed, provided that it has at least one mapped table.

A list of the mapped source tables in the specified database appears. The following image shows a list of the mapped source tables in the DB_Source database:



Tip: To display all source tables and databases, including those that are unmapped, select **Show all objects**. To display only the source tables that do not have Change Data Capture enabled, select **Show only the tables for which CDC is not set**. To filter the list of tables, enter the first few letters of a table name in the *Table Filter* field. For case-sensitive filtering, enclose the filter value in double quotation marks.

Important: If the source schema contains thousands of tables or the Data Replication Console runs in a local area network that is remote to the database server network, selecting **Show all objects** might significantly increase metadata caching time.

3. In the **CDC** column, select the check box for each source table for which you want to enable Change Data Capture, or clear the check box for each source table for which you want to disable Change Data Capture.

   If the source database is Microsoft SQL Server Standard Edition, Data Replication prompts you to confirm the change because you must restart the SQL Server instance to update the Change Data Capture settings. For more information about restarting the instance, see "Editing Microsoft SQL Server Instance Settings" on page 135.

   Tip: To enable Change Data Capture for all of the source tables, click **Select All**. To disable Change Data Capture for all of the source tables, click **Clear All**. To revert to the Change Data Capture settings that appeared when you first opened the **Manage SQL Server Change Data Capture Settings** dialog box, click **Reset**. To refresh the view with any Change Data Capture changes that were made directly in the source database or from the Server Manager CLI, click **Refresh**.

Data Replication generates Transact-SQL statements for executing the SQL Server stored procedures that enable or disable Change Data Capture for the source tables and the database that contains them. Data Replication uses the following syntax for the Change Data Capture statements:

```
use [database_name];
EXECUTE sys.sp_cdc_enable_db;
use [database_name];
declare @table_name nchar(characters); set @table_name=(SELECT table_name from
sys.objects where object_id=object_id) ; execute sys.sp_cdc_enable_table
@schema_name = N'dbo' , @source_name = @table_name , @role_name = NULL,
@capture_instance = N'dbo_table_name_sequence_ID';
```

```
use [database_name];
declare @table_name nchar(characters); set @table_name=(SELECT table_name from
sys.objects where object_id=object_id) ; execute sys.sp_cdc_disable_table
@schema_name = N'dbo' , @source_name = @table_name , @role_name = NULL,
@capture_instance = N'dbo_table_name_sequence_ID';
```

The sys.sp_cdc_enable_db procedure enables Change Data Capture for the database that contains the selected tables. The sys.sp_cdc_enable_table procedure enables Change Data Capture for a selected table, and the sys.sp_cdc_disable_table procedure disables Change Data Capture for a selected table.

4. To save the Transact-SQL statements to a script file, perform the following substeps:
   a. Click **Save**.

      The **Save** dialog box appears.

   b. Enter a path and file name. Then click **Save**.

      A message confirms that the script file was saved successfully.

   c. Click **OK**.

   You can use the saved script file to run the SQL statements later. If you do not have the authority to run the SQL statements or manage Change Data Capture settings, give the script file to a database administrator who has the required authority level.
5. To run the SQL statements before exiting the dialog box, click **Run**.
6. Click **Close**.

If you enabled Change Data Capture for any unmapped source tables, map these tables on the **Map Tables** tab to include them in the replication configuration.

Related Topics:

- "Editing Microsoft SQL Server Instance Settings" on page 135
- " Coordination with Microsoft SQL Server Change Data Capture" on page 70

Managing Oracle Supplemental Log Groups

For Oracle sources, you can use the Data Replication Console to add or remove the mapped source table columns that are in supplemental log groups. If you did not enable the Console to generate supplemental log groups when you saved the configuration, you can also add a supplemental log group.

To manage supplemental log groups in the Data Replication Console, the configuration must be open in Edit mode. To only view supplemental log groups, without making changes, the configuration can be open in Read mode.

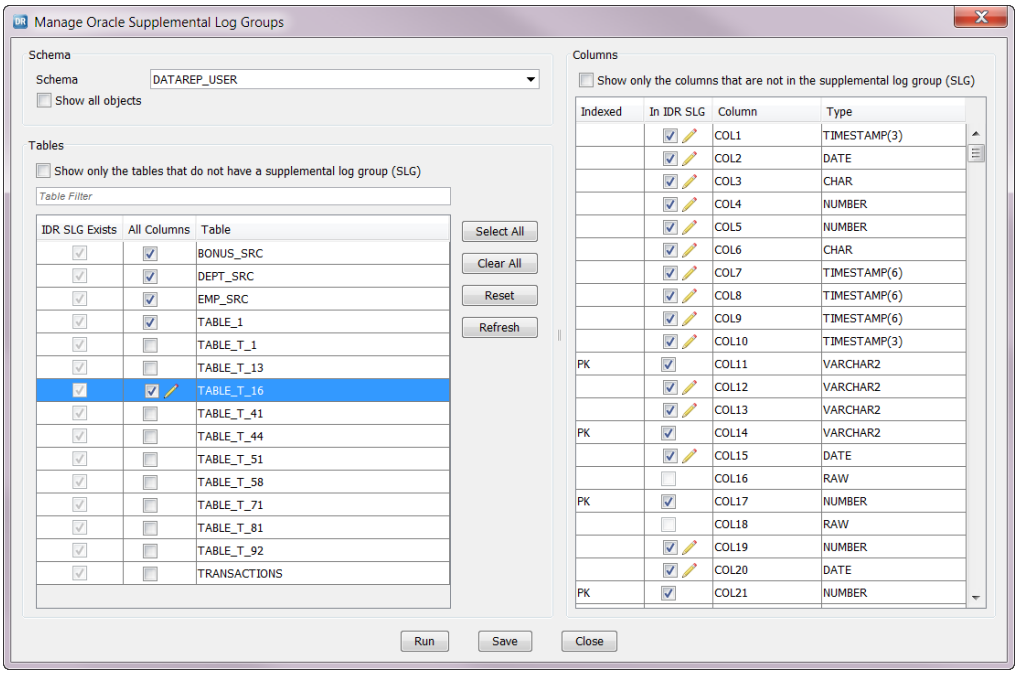
1. On the **Map Tables** tab, click the **Manage Database Supplemental Logging** icon button above the source table *Filter* list.

   The **Manage Oracle Supplemental Log Groups** dialog box appears.

2. In the **Schema** list, select the source schema name for the mapped tables for which you want to view or edit supplemental log group columns.

   Note: The **Schema** list displays only the schemas for which one or more source tables are mapped. By default, the source schema that is currently selected on the **Map Tables** tab is displayed, if it has at least one mapped table.

   A list of the mapped source tables that use the specified schema appears in the left list box. The following image shows the list of mapped source tables for the DATAREP_USER schema:



   Tip: To display all source tables and schemas, including unmapped tables and schemas, select **Show all objects**. To display only the source tables that do not have a supplemental log group, select **Show only the tables that do not have a supplemental log group (SLG)**. To filter the list of source tables, enter the first few letters of a table name in the *Table Filter* field. For case-sensitive filtering, enclose the filter value in double quotation marks.

   Important: If the source schema contains thousands of tables or the Data Replication Console runs in a local area network that is remote to the database server network, selecting **Show all objects** might significantly increase metadata caching time.

3. In the left list box, select the source table for which to view or manage the columns in the supplemental log group.

   The **Columns** list box on the right displays the table columns and indicates which columns are in the supplemental log group. The **Indexed** column indicates whether a column is part of an index or primary key. To display only the columns that are *not* in the supplemental log group, select **Show only the columns that are not in the supplemental log group (SLG)**.

   Note: Virtual columns in source tables do not appear in the **Columns** list box. You cannot add virtual columns to a supplemental log group.

4. To add all of the columns in the selected source table to the supplemental log group, select **All Columns** check box for the table in the left list box. To remove all of the columns in the selected source table from the supplemental log group, clear the **All Columns** check box.

   If you want to select or clear the columns individually in the **Columns** list box on the right, clear the **All Columns** check box for the table.

Attention: If a table has an existing supplemental log group that includes columns that are part of an index or primary key, clearing the **All Columns** check box deselects those columns. Informatica recommends that you always include columns that are part of an index or primary key in a supplemental log group to replicate change data accurately.

After you edit the **All Columns** check box, a pencil icon appears next to it.

5. In the **Columns** list box on the right, use the **In IDR SLG** check box to edit the columns that are in the supplemental log group:
   - To remove a column, clear the **In IDR SLG** check box.
   - To add a column, select the **In IDR SLG** check box.

   Note: The **In IDR SLG** check box is unavailable for columns that have the following attributes:
   - Columns that contain Oracle datatypes that Data Replication does not support. You cannot add these columns to a supplemental log group.
   - Columns that contain the following Oracle datatypes: BLOB, CLOB, LONG, LONG RAW, NCLOB, and RAW. You cannot add these columns to a supplemental log group.

   Tip: To select all eligible columns in all of the source tables, click **Select All**. To clear all columns in all of the source tables, click **Clear All**. To revert to the supplemental log group settings that appeared when you first opened the **Manage Oracle Supplemental Log Groups** dialog box, click **Reset**. To refresh the supplemental logging settings to get the latest changes from the source database or the Server Manager CLI, click **Refresh**.

   If a supplemental log group exists, Data Replication generates a pair of SQL statements for each table for which you change the columns. Data Replication uses the following syntax for the supplemental log group SQL statements:

   ```
   ALTER TABLE "schema_name"."table_name" ADD SUPPLEMENTAL LOG GROUP
   IDR_<object_ID_of_the_source_table>_<sequence_number> ("column_name","column_name")
   ALWAYS
   ALTER TABLE "schema_name"."table_name" DROP SUPPLEMENTAL LOG GROUP
   IDR_<object_ID_of_the_source_table>_<sequence_number>
   ```

   The ADD statement creates a new supplemental log group with the columns that you selected, and the DROP statement removes the previous supplemental log group if one exists.

   If a supplemental log group does not yet exist for a mapped source table, Data Replication generates only the ADD statement.

   Warning: If you edit the columns in the supplemental log group of a source table that is mapped in other configurations, make sure that you include all of the mapped columns from all of the configurations in the supplemental log group. Otherwise, errors might occur during replication processing. Include all table columns if you are not sure which columns to include.

6. To save the SQL statements to a script file, perform the following substeps:
   a. Click **Save**.

      The **Save** dialog box appears.

   b. Enter a file name and path and click **Save**.

      A message confirms that the script file was saved successfully.

   c. Click **OK**.

   You can use the saved script file to run the SQL statements later. If you do not have the authority to run the SQL statements or manage supplemental log groups, give the script file to a database administrator who has the authority to run the statements.

7. To run the SQL statements before exiting the dialog box, click **Run**.
8. Click **Close**.

If you added any columns in unmapped source tables to a supplemental log group, map these tables on the **Map Tables** tab to include them in the replication configuration.

Deploying Replication Configurations

Before you perform a full or partial deployment of a configuration to a local or remote Server Manager Main server, review the following concepts, requirements, and considerations.

Deployment Overview

In the Data Replication Console, you can deploy a replication configuration for which you are the owner to the local Server Manager Main server or to a remote Main server. If you are the idradmin user, you can deploy any replication configuration.

In the Data Replication Console, the term *full deployment* refers to deploying an entire configuration to the local Server Manager Main server or a remote Main server. You must perform a full deployment the first time you deploy a configuration file. The term *partial deployment* refers to deploying only the configuration changes that have occurred since the last full deployment.

In the following common scenarios, perform a full deployment:

- Deploy a configuration from a test environment to a remote production environment.
- Deploy a configuration to the local Server Manager Main server to quickly generate multiple configuration files based on the original configuration.
- Deploy a configuration to the local Server Manager Main server to configure a replication job that replicates data from multiple sources to a single target.

The Data Replication documentation uses the following terms to refer to the configurations and environments that are involved in the deployment process:

- **Destination environment**. The environment to which you deploy a configuration or configuration changes. The deployment destination can be a remote production environment or a local environment that uses the same Server Manager Main server.
- **New configuration**. The copy of the original configuration in the destination environment.
- **Original configuration**. The configuration that you deploy from the original environment.
- **Original environment**. The environment from which you deploy the original configuration.

Related Topics:

- "Replication from Multiple Sources to a Single Target" on page 25
- "Conflict Resolution for Replicated Data" on page 41

Deployment Requirements

Before you deploy a configuration or configuration changes, ensure that the following requirements are met:

- For a remote deployment, ensure that all components in both environments use the same version of Data Replication. If one environment uses an earlier Data Replication version, upgrade that Data Replication installation.

- The mapped source and target tables must exist in the destination environment. Also, the source and target table definitions must be the same in both environments.
- You can replace the schema names from the original environment during the initial configuration deployment. However, do not replace a single schema name with multiple schema names or replace multiple schema names with a single schema name. When defining schema replacement rules, ensure that each schema in the original environment corresponds to a single schema in the destination environment.
  If the schemas are different, the new configuration in the destination environment includes only the column mappings that exist in the tables in both environments. In this case, the column positions in the new configuration might be incorrect. You can save the new configuration to update the column positions so that data can be replicated accurately in the destination environment.
- If Oracle source and target tables contain any columns defined as UNUSED, these columns must be the same in both environments. Unused columns are defined by executing the ALTER TABLE *table* SET UNUSED *column* statement.

Deployment Considerations

The following considerations apply to deploying replication configurations:

- For a full deployment, you do not need to stop replication tasks in either the original or destination environment before performing the full deployment.
- If you perform a full deployment of a configuration with an Oracle source that includes tablespaces or columns that are encrypted with Oracle TDE, the Oracle wallet password and internal master key IDs are not deployed from the original configuration. After deployment, if you open the Oracle wallet and generate new internal master keys for the new configuration in the destination environment, Data Replication will retain these keys during future partial deployments of the configuration changes.
- To update a deployed configuration later, use one of the following strategies:
  - Update the configuration directly in the destination environment.
  - Update the configuration in the original environment and then use the **Partial deploy** option to deploy only the configuration changes to the destination environment. When you perform a partial deployment of configuration changes, stop all of the replication tasks that use the configuration, deploy the configuration changes, and then start the replication tasks again.

  Use the same update strategy for all subsequent updates of the configuration.
- Data Replication deploys only the replication settings of a configuration and not any of the internal Data Replication processing information for the configuration, which should not be modified. Depending on the deployment type, Data Replication handles the internal processing information in one of the following ways:
  - For an initial full deployment of a configuration, Data Replication creates a copy of the configuration in the destination environment. This copy does not contain the internal information that the replication tasks added to the configuration in the original environment. You can use the new configuration to start replication in the destination environment.
  - For a partial deployment of configuration changes, Data Replication does not modify the internal information that the replication tasks previously added to the configuration in the destination environment. You can resume all of the replication jobs with the updated configuration in the destination environment.

Performing a Full Deployment of a Configuration File

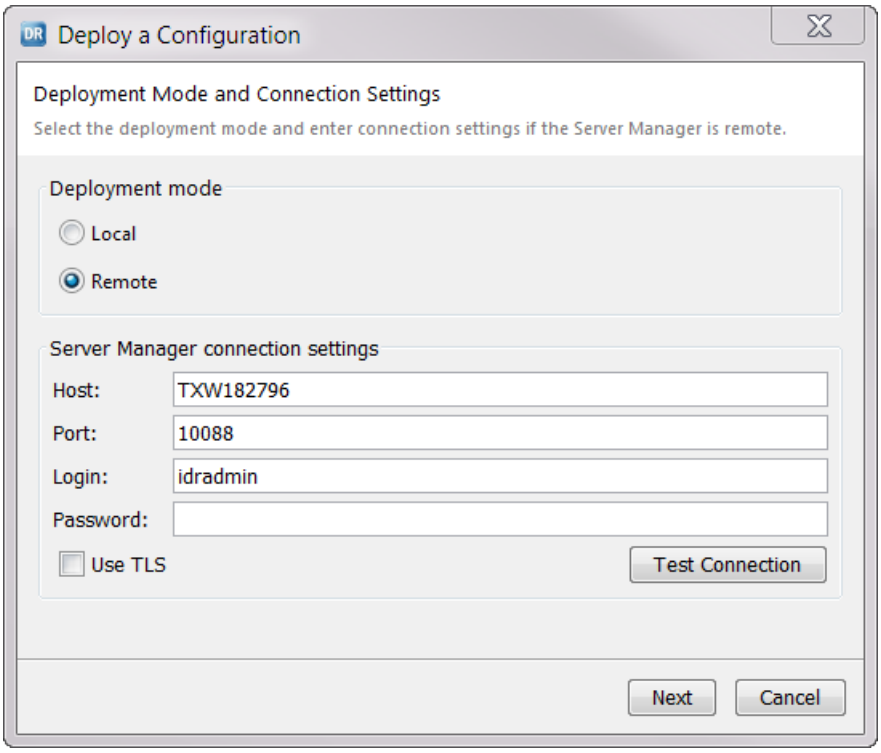
You can perform a full deployment of a configuration for which you are designated the owner to the local Server Manager Main server or to a remote Server Manager Main server.

You must perform a full deployment the first time you deploy a configuration locally or remotely. Thereafter, you can perform a partial deployment of only configuration changes or perform another full deployment of the configuration.

Important: You must close any open replication configuration in the Data Replication Console before you start the deployment procedure.

1. Connect to the Server Manager Main server on the system from which you are deploying the configuration.
2. On the **Server Manager** tab > **Configs** view, select an existing configuration.
3. Click the **Deploy** icon button on the Replication Configurations toolbar.

   The **Deploy a Configuration** wizard starts. The first **Deployment Mode and Connection Settings** page appears.



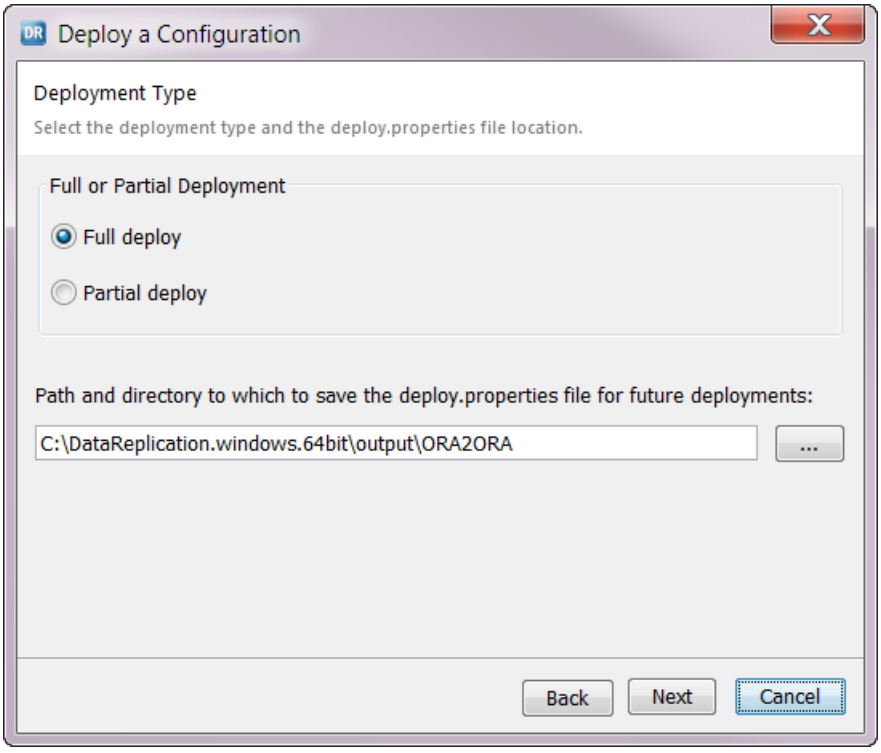
4. Select **Local** to deploy the configuration changes to the local Server Manager environment, or select **Remote** to deploy the configuration changes to another Server Manager environment.
5. For a remote deployment, specify the following Server Manager connection settings:
   - **Host**. The host name or IP address for the Server Manager instance.
   - **Port**. The port number. Valid values are integers from 1 through 65535.
   - **Login**. The user name.
   - **Password**. The password for the specified user name.
   - **Use TLS**. Select this check box if the remote Main server uses HTTPS connections.

Then click **Test Connection**. Data Replication verifies that the remote Server Manager instance is available and that the user credentials are valid.

6. Click **Next**.

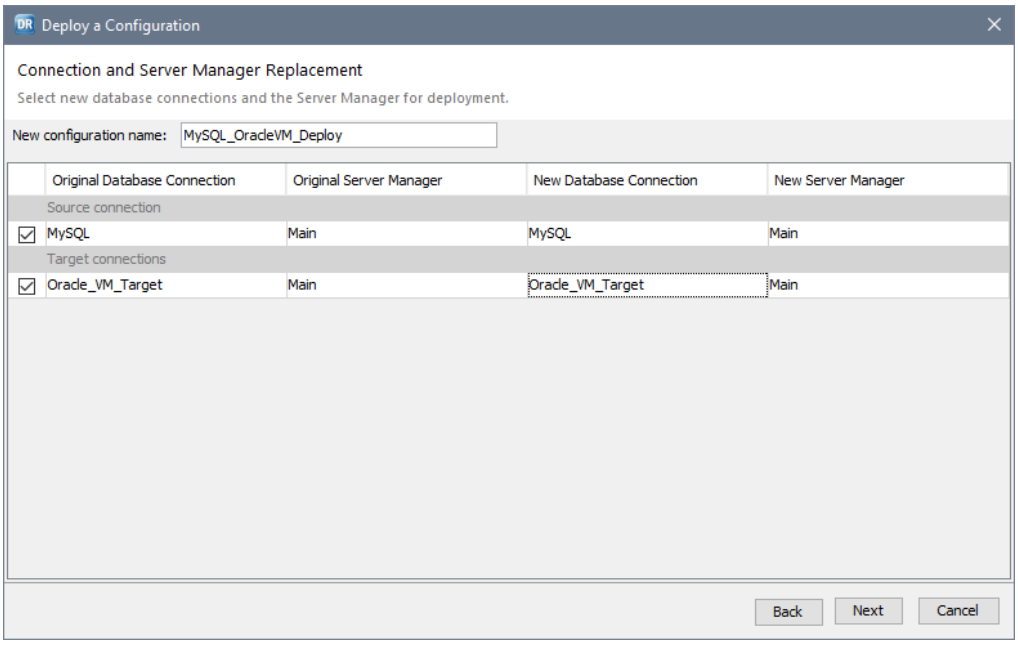
   If you are performing a remote deployment, the Data Replication Console displays a prompt that asks you to confirm termination of all other Data Replication sessions for the specified user. Click **Yes** to continue.

   The **Deployment Type** page appears.



7. Select **Full deploy**.
8. Enter the directory where the generated deploy.properties file will be saved at the end of the deployment process, or click the **Browse** button to browse to this directory. Then click **Next**.

   The **Connection and Server Manager Replacement** page appears.



Note: The **New configuration name** field displays the original configuration name.

9. In the **New configuration name** field, enter a new name for the configuration copy.

   Note: Configuration names are case-insensitive.

10. In the **New Database Connection** column, select the source and target connections for the new configuration. By default, only one source connection and one target connection for the configuration are selected. If the configuration has multiple targets and you want to deploy multiple target connections, select the check box to the left of each additional target connection.

    Notes:

    - Use source and target database connections that include an actual host name or IP address. Data Replication cannot deploy a configuration with a connection that uses "localhost" as the host name or uses 127.0.0.1 as the IP address.
    - If an Oracle source database in the original environment uses an Oracle ASM connection, the source database in the destination environment must also use an Oracle ASM connection.
    - If the **New Database Connection** field for the target connection is blank, no connections have been defined for the target Server Manager.

11. In the **New Server Manager** column, select the Server Manager for the new source and target connections, and click **Next**.

    The **Schema Replacement** page appears.

12. If you want to replace the original schema names with other schema names in the new configuration, perform the following substeps:

    a. Click the **New** icon button on the toolbar.

       The **New Schema Replacement** dialog box appears.

    b. In the **Connection** field, select the source or target database connection in the original configuration that includes the schema name you want to replace.

       The **Original Schema** field displays the schema name for the connection you selected.

c. In the **New Schema** field, select the replacement schema name for the connection in the new configuration.

d. Click **Save**.

The connection, original schema, and new schema appear in the **Schema Replacement** list.

e. Repeat steps a through d for each source or target database connection in the original configuration that includes a schema you want to replace during deployment.

Notes:

- For replication configurations that have multiple targets, you can define a schema replacement pair only for the primary target. After you deploy the configuration, change the schema names of the secondary targets in the new configuration on the **Routing** tab > **Schema Name** view.
- If you define schema replacement pairs for schemas that contain tables with identical names, ensure that the schema replacement pairs do not result in the following types of configuration problems:
  - One table is incorrectly mapped to multiple tables.
  - Multiple tables are incorrectly mapped to one table.
  - A table is mapped to itself.

  If any of these problems exist, the Data Replication Console deploys an invalid configuration to the destination environment.
- To edit a schema replacement pair, click the **Edit** icon button on the toolbar.
- To delete a schema replacement pair, click the **Delete** icon button on the toolbar.
- For Cloudera, Flat File, Hortonworks, and Kafka targets, the target schema name must match the source schema name. If you replace the source schema name, the target schema must also be changed.

13. Click **Next**.

The **Extract Range** page appears.

14. To disable change data capture from online logs, clear one of the following options, depending on the source type:
    - For DB2 and Oracle sources, clear **Read from online logs**.
    - For Microsoft SQL Server sources, clear **Read from online transaction logs**.

    By default, Data Replication captures change data from online logs for DB2, Microsoft SQL Server, and Oracle sources. For MySQL sources, Data Replication captures change data from binary logs only.
15. To edit the default location of the database log files that is displayed in the **Path** column and to filter the log files, perform the following substeps:
    a. Click **Add Directory**.
    b. To indicate the location of the log files, select one of the following options:
       - **File system**. Use the location of the log files that you specify on the file system of the computer where the source Server Manager runs or on a network mapped share, such as NFS or Samba. This option is the only option available for MySQL sources.
       - **Oracle ASM**. For Oracle sources, use the location of the archived log files in ASM.
       - **Default archive logs**. For DB2, Microsoft SQL Server, and Oracle sources, search for the log files in the database-specific location.

       This option determines what is displayed in the directory tree.

c. Specify the directory that Data Replication scans for the database log files in one of the following ways:

- In the **User-defined path** field, type the directory path and press Enter.
- In the directory tree, select the directory that contains the log files.

d. To enable Data Replication to recursively search for log files in subfolders of the specified directory for DB2, Microsoft SQL Server, or Oracle sources, select **Scan subfolders**.

e. To filter the log files that Data Replication reads for DB2, Microsoft SQL Server, or Oracle sources, enter a mask for the log file names in the **Filter files** field. To filter the binary log files that Data Replication uses for MySQL sources, enter a mask for the base name of the binary log files in the **Base Bin Log File Name** field.

You can use one or more of the following wildcard characters in the mask:

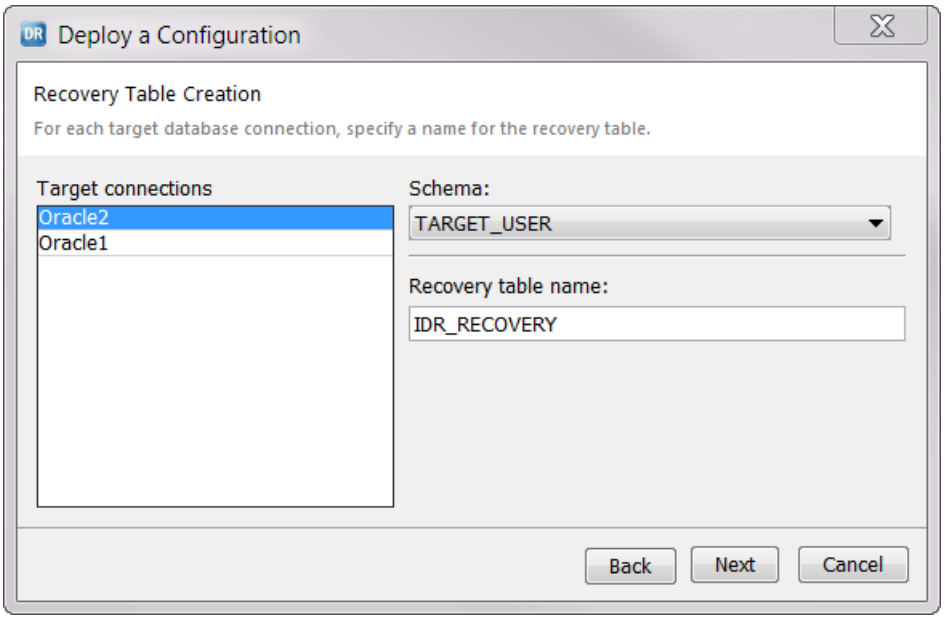
- An asterisk (*) to match all characters
- A question mark (?) to match any single character
- Square brackets ([ ]) to match only one out of several characters

For example, to read all files with the .bak extension, enter *.bak.

Important: The **Filter files** or **Base Bin Log File Name** field cannot be empty.

16. Click **Next**.

The **Recovery Table Creation** page appears if recovery tables are enabled for the configuration.



17. Enter information for creating a recovery table for the target in the destination environment. Perform the following substeps:

a. In the **Target connections** list, select a target connection.

b. In the **Schema** list, select the schema for the table.

If the target is a Microsoft SQL Server database, select the database name in the **Database** list, and then select the **Schema** value.

c. In the **Recovery table name** field, enter a name for the recovery table.

The default name of the recovery table is IDR_RECOVERY.

18. Click **Next**.

The **Deployment Summary** page displays a summary of the deployment settings.

19. To change a deployment setting, click **Back** to return to the previous wizard page where you can edit the setting. After you confirm that all of the deployment settings are correct, click **Deploy**.

    A progress dialog box indicates the status of deployment processing. When processing ends, the **Deployment Finished** page displays a status of Success or Failure. The page also lists the original and new configuration names and Server Managers.

20. Click **Close**.

    Data Replication generates a deployment log file that contains error, warning, and informational messages that were issued during deployment. The deployment.log file is located in the *DataReplication_installation*/logs directory.

For DB2, Microsoft SQL Server, and Oracle sources, the deployment procedure does not enable supplemental logging for the source tables in the destination environment. Before you run data replication in the destination environment, enable supplemental logging for these source types manually.

Related Topics:

- "Performing a Partial Deployment of Configuration Changes" on page 356

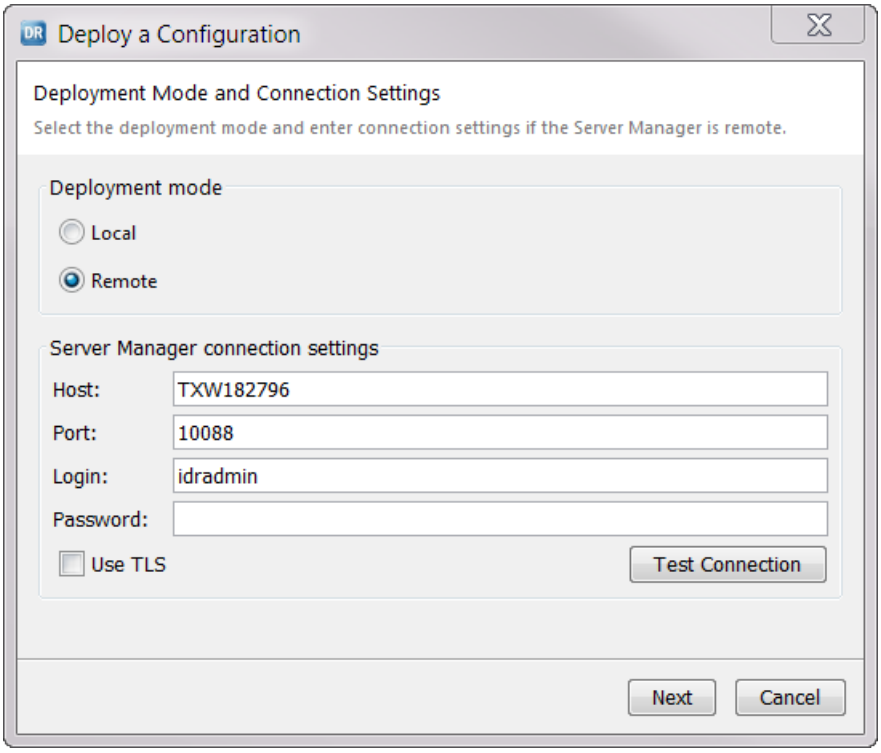
Performing a Partial Deployment of Configuration Changes

After you perform an initial full deployment of a replication configuration, you can deploy the configuration changes locally or to another Server Manager Main server.

Before performing a partial deployment, stop all active replication tasks that use the configuration.

1. Connect to the Server Manager Main server on the system from which you are deploying configuration changes.
2. On the **Server Manager** tab > **Configs** view, select the configuration that you want to deploy and click the **Deploy** icon button on the Replication Configurations toolbar.

The **Deploy a Configuration** wizard starts. The first **Deployment Mode and Connection Settings** page appears.

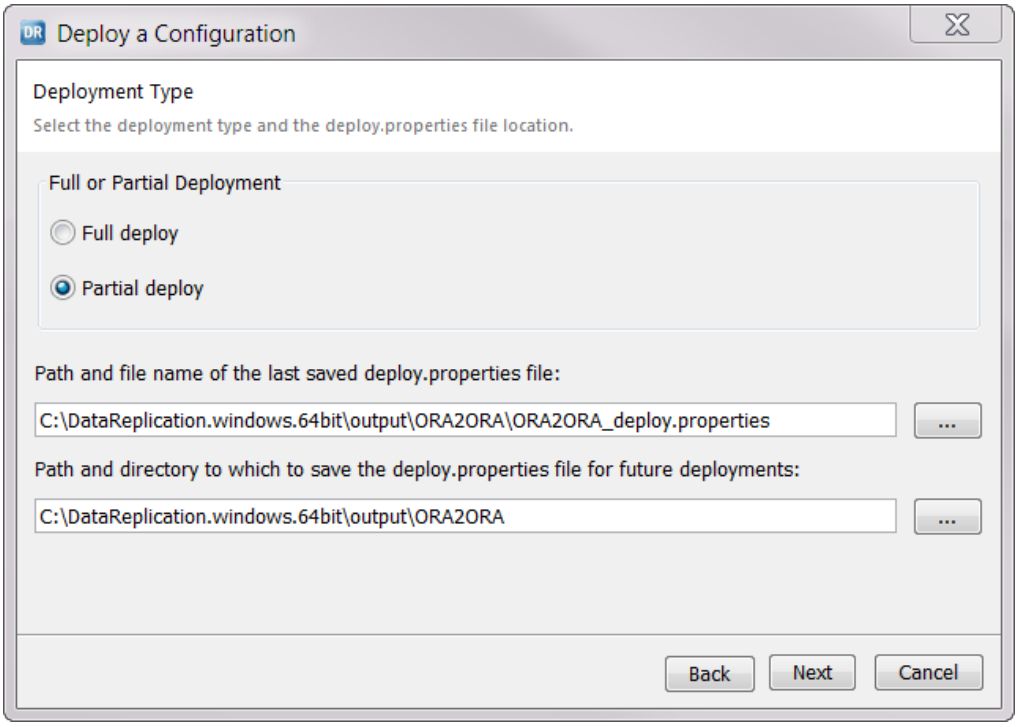


3. Select **Local** to deploy the configuration changes to the local Server Manager environment, or select **Remote** to deploy the configuration changes to another Server Manager environment.
4. For a remote deployment, specify the following Server Manager connection settings:
   - **Host**. The host name or IP address for the Server Manager instance.
   - **Port**. The port number. Valid values are integers from 1 through 65535.
   - **Login**. The user name.
   - **Password**. The password for the specified user name.
   - **Use TLS**. Select this check box if the remote Main server uses HTTPS connections.

   Then click **Test Connection**. Data Replication verifies that the remote Server Manager instance is available and that the user credentials are valid.
5. Click **Next**.

   If you are performing a remote deployment, the Data Replication Console displays a prompt that asks you to confirm termination of all other Data Replication sessions for the specified user. Click **Yes** to continue.

The **Deployment Type** page appears.



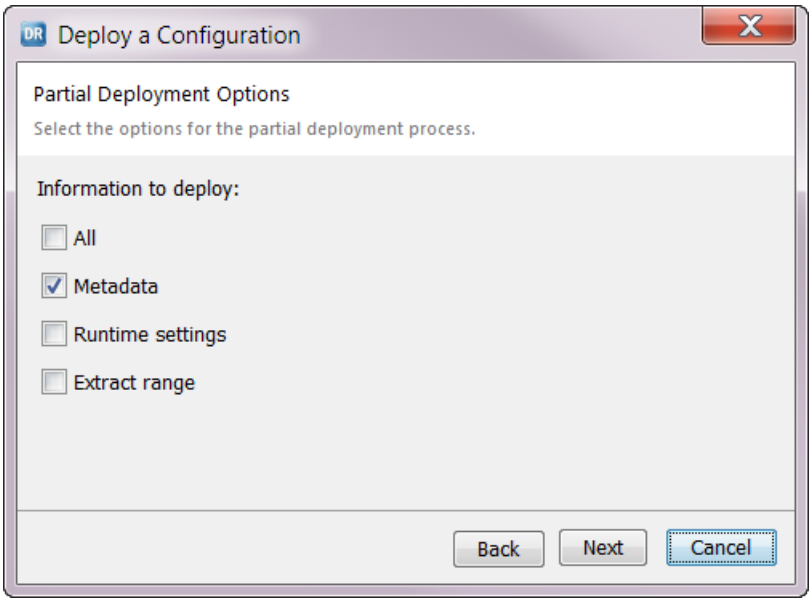
6. Select **Partial deploy**.
7. Enter the path and file name of the deploy.properties file that you used for the last full deployment of this configuration, or click the **Browse** button to browse to this file.

   Data Replication uses the settings in this file to configure the partial deployment.

   Warning: Data Replication does not validate that the deploy.properties file matches the configuration that you are redeploying. Data Replication validates only that the properties specified in the file are consistent with the configuration. If you select the wrong deploy.properties file, the configuration changes will not be deployed correctly, which can cause configuration inconsistencies or corrupt the configuration.

   If you do not specify a deploy.properties file, Data Replication will display a warning and prompt you to specify all of the deployment settings again.
8. Enter the path and directory to which you want to save the new deploy.properties file that Data Replication generates at the end of this partial deployment process, or click the **Browse** button to browse to the directory. Then click **Next**.

The **Partial Deployment Options** page appears.



9. To indicate the type of configuration changes to deploy, select one or more of the following options:
   - **All**. Deploys all configuration changes to the new configuration in the destination environment, including changes to metadata, runtime settings, and extract range settings. If you select this option, you cannot select any of the other options.
   - **Metadata**. Deploys only metadata changes to the new configuration in the destination environment. Metadata changes include table and column mappings and routing settings.
   - **Runtime settings**. Deploys only changes that were made on the **Runtime Settings** tab to the new configuration in the destination environment. However, this option does not deploy changes from the **Runtime Settings** > **Calculated Columns** view.
   - **Extract range**. Deploys only changes that were made on the **Extract Range** tab to the new configuration in the destination environment.

   Default is **Metadata**.
10. Click **Next**.

    The **Connection and Server Manager Replacement** page appears. This page displays the settings specified in the deploy.properties file that you selected. You cannot edit any of the settings on this page.
11. Click **Next**.

    The **Schema Replacement** page appears.
12. If you want to replace the original schema names with other schema names in the new configuration, perform the following substeps:
    a. Click the **New** icon button on the toolbar.

       The **New Schema Replacement** dialog box appears.
    b. In the **Connection** field, select the source or target database connection in the original configuration that includes the schema name you want to replace.

       The **Original Schema** field displays the schema name for the connection you selected.
    c. In the **New Schema** field, select the replacement schema name for the connection in the new configuration.

d. Click **Save**.

The connection, original schema, and new schema appear in the **Schema Replacement** list.

e. Repeat substeps a through d for each source or target database connection in the original configuration that includes a schema that you want to replace during deployment.

Notes:

- For replication configurations that have multiple targets, you can define a schema replacement pair only for the primary target.
- If you define schema replacement pairs for schemas that contain tables with identical names, ensure that the schema replacement pairs do not result in the following types of configuration problems:
  - One table is incorrectly mapped to multiple tables.
  - Multiple tables are incorrectly mapped to one table.
  - A table is mapped to itself.

  If any of these problems exist, the Data Replication Console deploys an invalid configuration to the destination environment.
- To edit a schema replacement pair, click **Edit** on the toolbar.
- To delete a schema replacement pair, click **Delete** on the toolbar.

13. Click **Next**.

The **Extract Range** page appears.

14. To edit the default location of the database log files that is displayed in the **Path** column and to filter the log files, perform the following substeps:

a. Click **Add Folder**.

b. To indicate the location of the log files, select one of the following options:

- **File system**. Use the location of the log files that you specify on the file system of the computer where the source Server Manager runs or on a network mapped share, such as NFS or Samba.
- **Oracle ASM**. For Oracle sources, use the location of the archived log files in ASM.
- **Default archive logs**. For DB2, Microsoft SQL Server, and Oracle sources, search for the log files in the database-specific location.

This option determines what is displayed in the directory tree.

c. Specify the directory that Data Replication scans for the database log files in one of the following ways:

- In the **User-defined path** field, enter the path and directory name, and press Enter.
- In the directory tree, select the directory that contains the log files.

d. To enable Data Replication to recursively search for log files in subfolders of the specified directory for DB2, Microsoft SQL Server, or Oracle sources, select **Scan subfolders**.

e. To filter the log files that Data Replication reads for DB2, Microsoft SQL Server, or Oracle sources, enter a mask for the log file names In the **Filter files** field. To filter the binary log files that Data Replication uses for MySQL sources, enter a mask for the base name of the log files in the **Base Bin Log File Name** field.

You can use one or more of the following wildcard characters:

- An asterisk (*) to match all characters
- A question mark (?) to match any single character

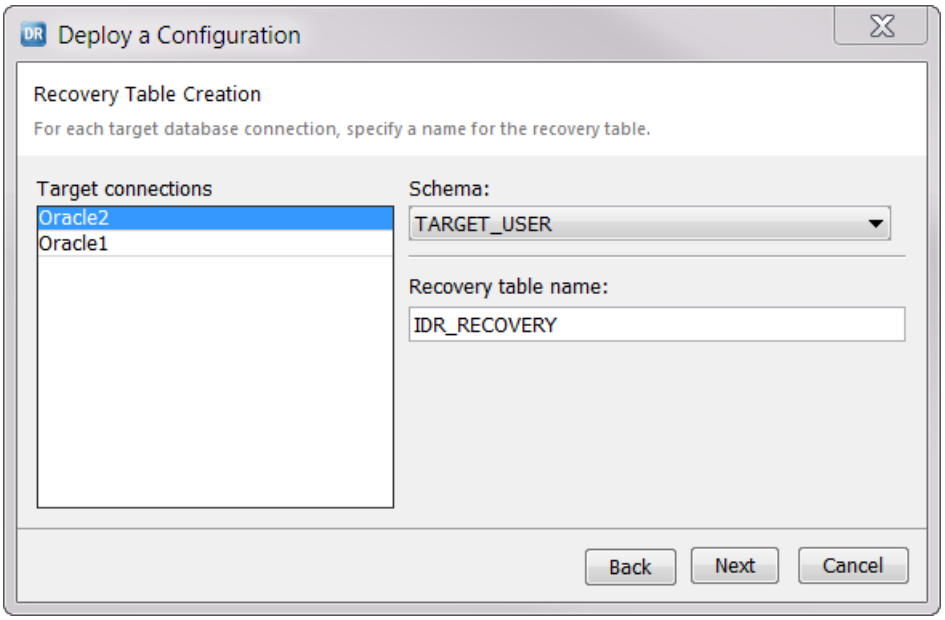
- Square brackets ([ ]) to match only one out of several characters

For example, to read all files with the .bak extension, enter *.bak.

Important: The **Filter files** or **Base Bin Log File Name** field cannot be empty.

15. Click **Next**.

The **Recovery Table Creation** page appears if recovery tables are enabled for the configuration.



16. To enter information for creating a recovery table for the target in the destination environment, perform the following substeps:
    a. In the **Target connections** list, select a target connection.
    b. In the **Schema** list, select the schema for the table.

       If the target is a Microsoft SQL Server database, select the database name in the **Database** list, and then select the **Schema** value.
    c. In the **Recovery table name** field, enter a name for the recovery table.

       The default name of the recovery table is IDR_RECOVERY.
17. Click **Next**.

    The **Deployment Summary** page displays a summary of the deployment settings.
18. To change a deployment setting, click **Back** to return to the previous wizard page where you can edit the setting. After you confirm that all of the deployment settings are correct, click **Deploy**.

    A progress dialog box indicates the status of deployment processing. When processing ends, the **Deployment Finished** page displays a status of Success or Failure. The page also lists the original and new configuration names and Server Managers.
19. Click **Close**.

    Data Replication generates a deployment log file that contains error, warning, and informational messages that were issued during deployment. The deployment.log file is located in the *DataReplication_installation*/logs directory.
20. If you stopped any replication tasks earlier to deploy configuration changes, start the tasks again with the updated configuration.

The deployment procedure does not enable supplemental logging for DB2, Microsoft SQL Server, or Oracle source tables in the destination environment. You must enable supplemental logging for the new tables

before you run data replication in the destination environment if both of the following conditions apply to the partial deployment:

- You selected the **All** or **Metadata** option on the **Partial Deployment Options** page.
- The partial deployment included new tables that were created after the previous deployment of the configuration.

For more information, see "Managing Database Supplemental Logging" on page 342.

Related Topics:

- "Performing a Full Deployment of a Configuration File" on page 351

Generating a Reverse-Replication Configuration

From the Data Replication Console, you can generate a reverse-replication configuration to use for bidirectional replication or for failover to a target database.

1. On the **Server Manager** tab > **Configs** view, select the configuration for which you want to generate a reverse-replication configuration.
2. To generate a reverse configuration, use one of the following procedures:
   - Click the **Generate Reverse Configuration** toolbar button. Then click **OK**.
   - Right-click the configuration row and click **Generate Reverse Configuration**. Then click **OK**.
   - Open the configuration and click **File** > **Generate Reverse Configuration**.
3. Enter a name for the reverse-replication configuration and click **OK**.

   Configuration names are not case-sensitive. They can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates configuration names that are longer than 100 characters.

   The Data Replication Console generates a configuration and opens it in Edit mode.

   Note: If Data Replication creates a revision of the original configuration and increments the revision number after you open the configuration, the Data Replication Console cannot generate a reverse-replication configuration. To generate a reverse-replication configuration that uses the latest revision of the original configuration, open the original configuration again.
4. Configure replication settings on the **Extract Range** tab and the **Runtime Settings** tab > **Advanced Settings** view.
5. To create a recovery table, perform the following substeps:
   a. Click the **Runtime Settings** tab > **Advanced Settings** view.
   b. Double-click the apply.recovery_table parameter row.
   c. Specify a recovery table name that includes the schema or owner name.
   d. Click **OK**.

   When you start the Applier later, it creates a recovery table on the target with the specified name.
6. Click the **Save** toolbar button to save the configuration.

After you generate a reverse-replication configuration, the Data Replication Console does not keep it synchronized with the original configuration. If you edit the original configuration, you must manually add the changes to the corresponding reverse-replication configuration. If you enabled DDL replication for one of the

configurations, and DDL changes occur, you must manually add the appropriate changes to the reverse-replication configuration.

Related Topics:

- "Configuring Bidirectional Replication" on page 307

Exporting a Configuration File

You can export a configuration that is stored in the SQLite database in XML and SQLite database format to a specified directory.

If you use Informatica Fast Clone with the Fast Clone Server option, you can copy the exported configuration XML file for Fast Clone Server use. From the Fast Clone **Runtime Settings** > **DatabaseSync Integration** view, you must enable Fast Clone integration with Data Replication and then point to the exported configuration XML file.

1. On the **Server Manager** tab > **Configs** view, select the configuration that you want to export and then click the **Export** icon button on the Replication Configurations toolbar.

   The **Export Configuration 'config_name'** dialog box appears.

   Note: The configuration does not need to be open for you to export the configuration file.
2. Select one of the following options to indicate the system to which you want to export the configuration:
   - **Server Manager Main server system**. Exports the configuration to the computer where the Server Manager Main server runs.
   - **Data Replication Console system**. Exports the configuration to the computer where the Data Replication Console runs.
3. In the **Path to the destination folder** field, enter the path to the folder to which to export the configuration, or click the **Browse** button to browse to the folder.

   Note: If you enter a destination folder that does not yet exist, the Data Replication Console creates the folder at the path you specify, provided that the path and parent folders are valid. Destination folder names cannot contain spaces.
4. Click **Export**.

   The **Report of Operation** dialog box displays the status of each export operation. Statuses are: success, failed, or skipped. The **Total** row indicates whether the overall configuration file export succeeded.

   Notes:
   - Data Replication copies information from source and target connections to the XML configuration file and the configuration SQLite database.
   - If you export a configuration that has multiple targets, the XML configuration file and the configuration SQLite database include connection information only for the primary target. Connection information for the secondary targets is lost. However, because Data Replication exports the Applier and Send File SQLite databases, the other target information in these databases is available for each target.
5. If the export operation succeeded, click **Close**. Otherwise, resolve any failures and try to export the configuration again.

Importing a Configuration File

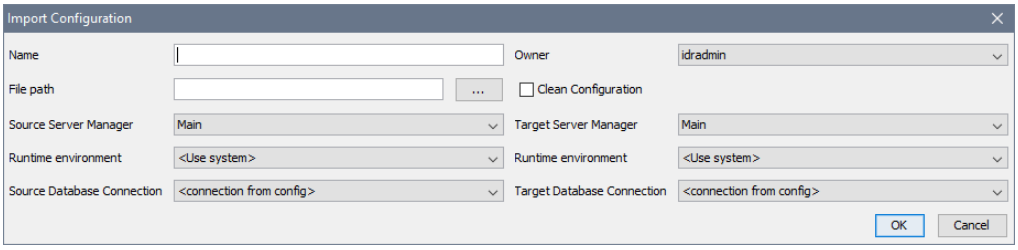
You can import a configuration file in XML or SQLite database format for Server Manager use. The file can be from the local replication environment or from another replication environment.

For example, you can import a configuration .xml file that you created in a test environment for use in the production environment.

Note: If you import a configuration that has multiple targets, Data Replication skips information about the secondary targets and imports information only for the primary target.

1. On the **Server Manager** tab > **Configs** view, click the **Import** icon button on the Replication Configurations toolbar.

   The **Import Configuration** dialog box appears.



2. In the **Name** field, enter a name that you want to use for the imported configuration file.

   Configuration names are not case-sensitive. They can contain the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character. The Data Replication Console truncates configuration names that are longer than 100 characters.

3. In the **Owner** list, select the configuration owner.

   Note: If you are a regular non-administrative user and select idradmin as the owner, you will not be able to edit the configuration later.

4. For the **File path** field, click the **Browse** button to browse to the configuration file that you want to import.

5. In the **File System for Server Manager 'name'** dialog box, perform the following substeps:

   a. To filter the configuration files, enter a configuration name mask in the **Filter** field.

      You can use the asterisk (*) wildcard in the mask.

   b. Select one of the following options to indicate the format of the configuration files to display in the directory tree:

      - ***.xml files** to display only XML configuration files

        Important: On Solaris systems only, the directory from which you import the XML configuration file must also contain the config.xsd schema file.

      - ***.db files** to display only SQLite configuration files

   c. In the directory tree, select the configuration file that you want to import.

      The **File** field displays the configuration file name.

   d. Click **OK**.

6. If you want to remove replication processing information for all replication tasks from the configuration, select the **Clean Configuration** check box.

7. Select the source server from the **Source Server Manager** list. Then select the environment variables list for the source from the **Runtime environment** list on the left.

8. Select the target server from the **Target Server Manager** list. Then select the environment variables list for the target from the **Runtime environment** list on the right.
9. In the **Source Database Connection** and **Target Database Connection** lists, select a connection that has been defined for the source database or target database in the destination environment or retain the default value of **<connection from config>**.

   If you use **<connection from config>**, the Console uses the imported connection information to generate a connection with a name that has the following format: <*configuration_name*>_source_connection or <*configuration_name*>_target_connection.
10. Click **OK**.

    The **Report of Operation** dialog box displays the status of each import operation. Statuses are: done, failed, or skipped. The **Total** row indicates whether the overall configuration import operation succeeded.
11. If the import succeeded, click **Close**.

    The configuration appears on the **Server Manager** tab > **Configs** view, and the Data Replication Console opens the imported configuration in Edit mode.

    Note: If the import failed, resolve any issues and try to import the configuration again.
12. Save the imported configuration to complete the import operation.

Cleaning Replication Processing Information for a Configuration

In the Data Replication Console, you can use the **Clean** option to remove replication processing information for all replication tasks that are associated with a replication configuration.

A clean operation removes the following information for the configuration:

- Intermediate files

  Note: The clean operation deletes intermediate files regardless of the values that are set for the Server Manager IntermediateFilesDeleteLatency and NotDeleteIntermediateFiles advanced properties. For more information about these properties, see "Editing Properties for the Main Server or a Subserver" on page 138.
- Checkpoint information
- Information about previous configuration revisions
- Internal Data Replication processing information that the replication tasks added to the configuration
- Information about open transactions and processed database logs

A clean operation does not remove replication configuration settings such as the source and target connection information, the recovery table connection information, and the location of the source database logs.

When you request a clean operation on a configuration, Data Replication backs up the existing configuration SQLite database files and intermediate files before starting clean processing, by default. The following information is backed up:

- All of the SQLite database files that are associated with the replication configuration
- All of the intermediate files that are associated with the replication configuration

You can clear the option for creating the backups if you do not want to retain this information. Data Replication stores the backups in the directory that is specified in the **Backups** field on the **Paths** tab of the Server Manager **Properties** dialog box. Data Replication creates a subdirectory for each clean operation using the following naming pattern: *<current_timestamp>_<configuration_name>*. By default, Data Replication stores each backup for 30 days. Data Replication checks the backup subdirectories hourly to remove any subdirectory for which the retention period has expired. You can use the Server Manager TimeToStoreBackups advanced property to change the number of days that the Server Manager Main server stores the backups.

After a clean operation, the Extractor and Applier tasks associated with the configuration appear as if they have never run. If no open transaction exists for the mapped source tables, the Extractor Start Point is set to the last SCN, log coordinate, or LSN value that the Applier processed for the configuration. If an open transaction exists for the mapped source tables, the Extractor Start Point is set to the last SCN, log coordinate, or LSN value for the BEGIN TRANSACTION operation.

Important: During clean processing, the Data Replication Console requires a source database connection to retrieve certain database information for the updated Extractor Start Point. If you clean a replication configuration when the source database connection is not currently available, the Data Replication Console does not update the Extractor Start Point value. In this case, you can set the Extractor Start Point value manually later.

The Applier Start Point is set to the last SCN, log coordinate, or LSN value that the Applier processed for the configuration.

Note: For Kafka targets, cleaning a configuration does not affect the checkpoints for apply processing that are recorded in the checkpoint file on the system where the Applier and Queue Adapter run.

Related Topics:

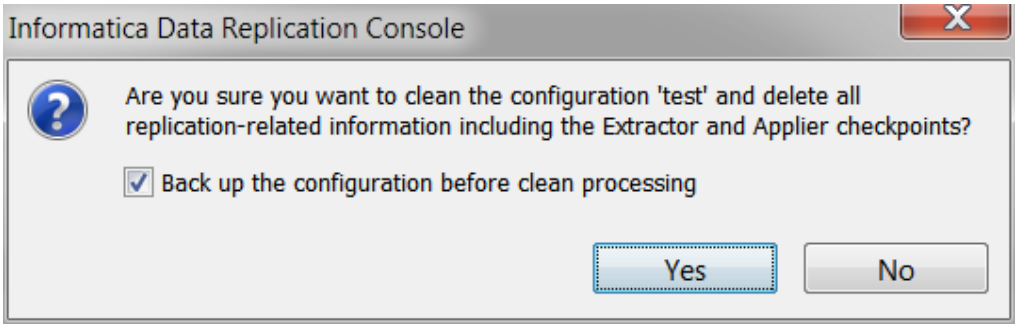
- "Start Point for the Extractor Task" on page 30

Performing a Clean Operation on a Configuration

You can remove replication processing information for all replication tasks that are associated with a replication configuration and optionally back up the existing configuration before starting clean processing.

1. On the **Server Manager** tab > **Configs** view, select the configuration that you want to clean.
2. Right-click the configuration row and click **Clean**, or click the **Clean** icon button on the Replication Configurations toolbar.

   The Data Replication Console prompts you to confirm the operation.



3. If you do not want to back up the existing configuration prior to cleaning it, clear the **Back up the configuration before clean processing** option.

   Important: Informatica recommends that you always back up the configuration to be cleaned. If a problem then occurs, the configuration information is retained in the backup for diagnostic use.

4. Click **Yes**.
5. For configurations that have a DB2 for Linux, UNIX, and Windows source, reset the db2.initial_lsn runtime parameter.

   The Extractor requires this step to correctly process the Extractor Start Point value.

Viewing Earlier Revisions of a Replication Configuration

You can view earlier revisions of a replication configuration. However, you cannot edit or save an earlier revision.

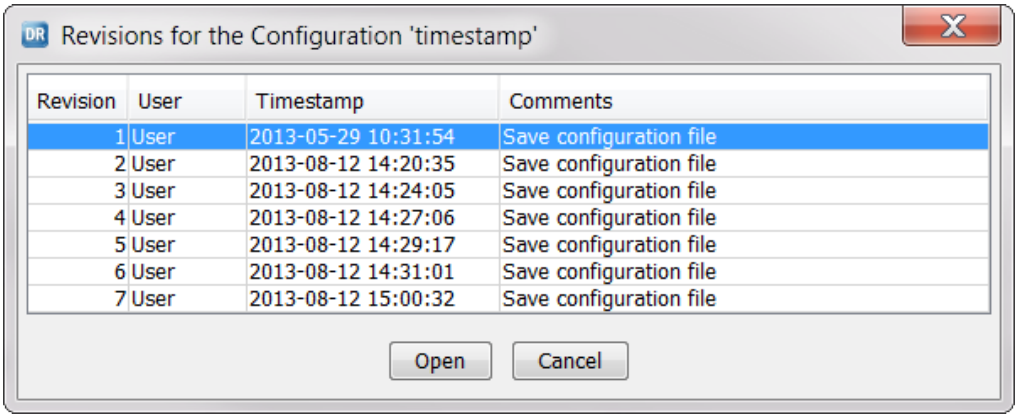
The Server Manager stores all of the revisions for the replication configurations in the following SQLite file: *DataReplication_installation*/configs/*configuration_name*.db. The Server Manager stores this file on the system where you run the Main server.

Data Replication increases the revision number for a replication configuration when any of the following events occur:

- You manually save a replication configuration, add filters to the replicated tables, or override the schema name for a configuration that has multiple targets.
- You import a replication configuration.
- The Extractor captures a DDL operation.

1. On the **Server Manager** tab > **Configs** view, select a configuration for which to view an earlier revision.
2. Click the **Revisions** toolbar button, or right-click the configuration row and click **Revisions**.

   The **Revisions for the Configuration 'config_name'** dialog box appears.



   The dialog box displays a row for each configuration revision. Each row contains the revision number, the user or component that made the revision, the date and time of the revision, and a brief description of the change.
3. Select the revision that you want to view.
4. Click **Open**.

   The Data Replication Console opens the configuration revision in Read mode.

Note: If you add or remove targets on the **Routing** tab across configuration revisions, the **Routing** tab shows the latest target settings regardless of which configuration revision you select.

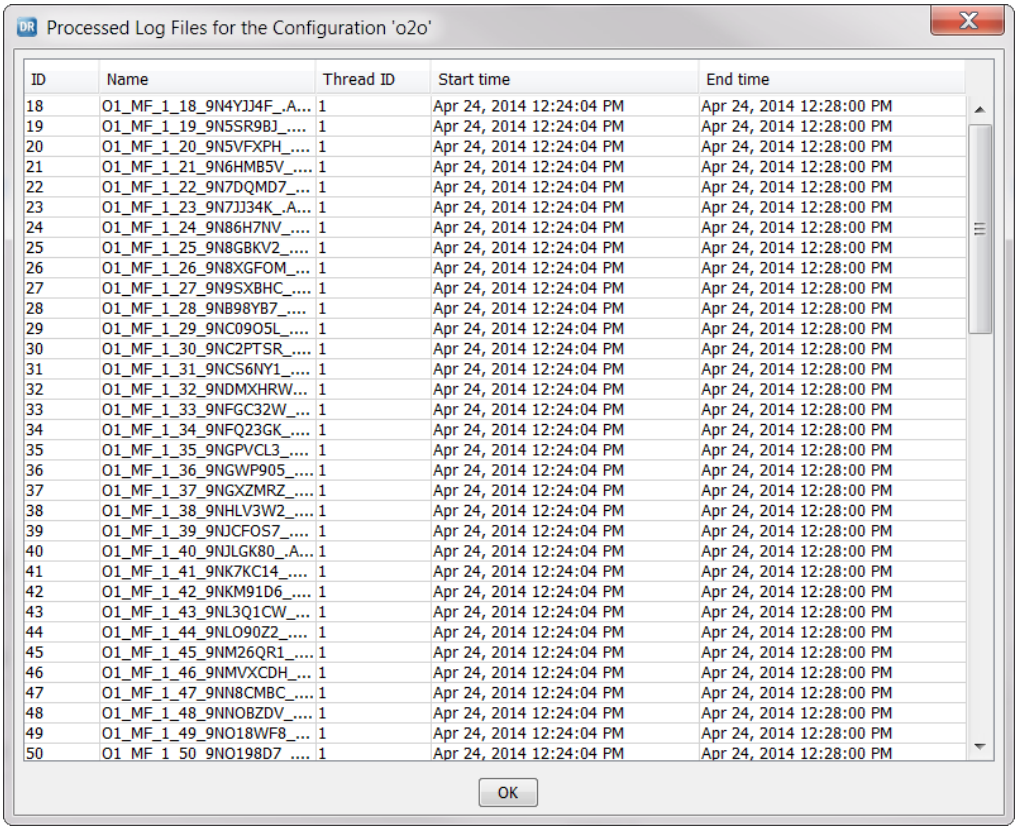
Viewing a List of Processed Database Logs

Data Replication stores information about the database logs that the Extractor processed for a configuration in the *DataReplication_installation*/configs/*config_name*.db file on the system where you run the Server Manager Main server. In the Data Replication Console, you can view the list of the processed database logs for a configuration.

This feature is useful if you need to safely purge archive or backup logs, without causing incomplete data replication.

1. On the **Server Manager** tab > **Configs** view, select a configuration for which to view a list of processed database logs.
2. Click the **Processed Logs** icon button, or right-click the configuration row and click **Processed Logs**.

   The following image shows the **Processed Logs for the Configuration 'config_name'** dialog box:



3. View the list of the database logs that the Extractor processed for the configuration.

   Note: For Microsoft SQL Server sources, the Data Replication Console displays only backup logs. For Oracle sources, the Data Replication Console displays both online and archived redo logs.

The following table describes the properties of processed logs:

Property	Description
ID	The sequence number of the processed log file.
Name	The name of the processed log file.
Thread ID	For Oracle sources, the unique identifier of a redo thread. For other sources, the unique identifier of the database.
Start time	The date and time when the Extractor opens an intermediate file that contains records from the processed log file.
End time	The date and time when the Extractor closes an intermediate file that contains records from the processed log file.

4. Click **OK** to close the dialog box.

CHAPTER 16

Handling Replication Environment Changes and Failures

This chapter includes the following topics:

Manually Changing the Source Table Structure After Running Data Replication

If you disabled replication of ADD COLUMN, ALTER COLUMN, and DROP COLUMN operations for a table from which Data Replication has captured change data, use the following procedures to manually add, modify, and drop the table columns and properly resume data replication afterwards.

Adding a Column

If you need to add a column that is not part of the primary key in a source table, use this procedure to make the change and resume replication.

1. Stop replication.
2. Back up your configuration file.
3. Add the column to the source table.
4. Add the column to the target table with the same default value as on the source.
5. In the Data Replication Console, open the configuration file.

   The file includes the new column.
6. On the **Map Columns** tab, map the new column on the source to the corresponding column on the target.
7. Save the configuration file.
8. Verify that the configuration file was saved correctly.

9. Continue replication.
10. If the column already has data in the source table, you must re-synchronize the source and target tables before starting data replication again.

Dropping a Column

If you need to drop a column that is not part of the primary key in a source table, use this procedure to make the change and resume replication.

1. Stop replication.
2. Drop the column from the source table.
3. Drop the column from the target table.
4. Switch logs and run an extraction cycle.

   This step ensures that all of the change records that reference the column are parsed correctly.
5. Back up your configuration file.
6. In the Data Replication Console, open the configuration file.

   The file no longer contains the column information and mapping for the dropped column.
7. Save the configuration file.
8. Continue replication.

Modifying a Column

If you need to increase the size or precision of a column that is not part of the primary key in a source table, use this procedure to make the change and resume replication. This procedure does not support modifications that require a datatype change.
Note: If you re-create a table, the new table has a different object ID. In this case, you must re-synchronize and map the table again in the Data Replication Console.

1. Stop replication.
2. Modify the column on the source table.
3. Modify the column on the target table.
4. Switch logs and run an extraction cycle.

   This step ensures that all of the change records that reference the column are parsed correctly.
5. Back up your configuration file.
6. In the Data Replication Console, open the configuration file.

   The file contains the updated column information.
7. Save the configuration file.
8. Continue replication.

Updating an Avro Schema for Kafka Targets After Running Data Replication

If the columns in a mapped source table definition change and you want to reflect that change in Kafka target messages, you must manually update the column mappings in the Data Replication Console and then delete the Avro schema cache file.

Before performing these steps, stop all replication tasks that are running for the configuration.

Note: If a column in a mapped source table is dropped, the Data Replication Console can automatically update the column mappings when you open the configuration. However, you still must delete the Avro schema cache file so that a new schema cache file can be re-created.

1. Open the configuration and switch to Edit mode.
2. Click the **Map Columns** tab.

   If source columns were dropped, verify that the Console automatically removed the mappings for those columns. Then skip to step 4.
3. If new source columns were added, perform the following steps to map them:
   a. In the **Source Table** list, select the table that contains the new column.

      The corresponding target table appears in the **Target Table** list.
   b. Select the new source column row.
   c. Select the target column row.
   d. Click **Map**.
4. Save the configuration.
5. Navigate to the directory to which Data Replication writes the cache files that contain Avro schema information.

   The default directory is *DataReplication_installation*/overflow. Another directory might have been specified in the apply.avro.avro_schema_cache_directory parameter on the **Runtime Settings** tab > **Advanced Settings** view. Avro schema cache file names use the following format: key*<key_number>*_*<source_owner>*_*<source_table_name>*.che.
6. Delete the Avro schema cache file that contains the source table name for which a column was added or removed.
7. Restart all replication tasks.

   The next time the Applier runs, Data Replication will create a new file that contains updated information about the Avro schema for the source table.

Adding Table Mappings Manually After Running Data Replication

If you disabled replication of CREATE TABLE operations in a configuration that has been used to run replication tasks, you can add table mappings to a configuration manually.

1. In the Data Replication Console, open the replication configuration that you want to edit.
2. Click the **Switch to Edit Mode** button on the toolbar.

3. The Data Replication Console displays a prompt that asks if you want to stop the replication tasks or keep them running.
   - Click **Yes** to stop the replication tasks.
   - Click **No** to keep the replication tasks running.

     Note: If the Extractor detects DDL changes in the source tables, it creates a new revision of the configuration. If a new revision is created while you are editing the configuration, you will not be able to save your changes. In this case, you must reload the configuration and redo the changes.
4. On the **Map Tables** tab, define mappings for the additional source tables that you want to replicate.
5. Click the **Save** button on the toolbar to save the configuration.

   The Data Replication Console detects that the newly mapped tables are not materialized and prompts you to run InitialSync.
6. Click **Yes** to run the InitialSync task for the added tables.

   If you chose to keep the replication tasks running, Data Replication starts the InitialSync task that runs simultaneously with the Extractor and Applier tasks. When the InitialSync task completes materializing the added tables, the Extractor and Applier start processing them.
7. If you stopped replication tasks prior to editing the configuration mappings, restart all of the replication tasks and schedules.

Resuming Replication After Upgrading a DB2 for Linux, UNIX, and Windows Source Database

Perform the following steps when you upgrade a DB2 for Linux, UNIX, and Windows source database:

1. Ensure that all of the database clients are disconnected from the DB2 source database.
2. Ensure that the Extractor finished processing the DB2 transaction logs.
3. Stop all replication tasks and schedules for the replication configurations that use the DB2 source database that you want to upgrade.
4. Upgrade the DB2 database server.

   For more information, see the IBM DB2 for Linux, UNIX, and Windows documentation.
5. Restart all of the replication tasks and schedules.

Resuming Replication After Upgrading a Microsoft SQL Server Source Database

If you upgrade a Microsoft SQL Server source database to a later version, perform the following steps to use your existing configurations to resume replication processing.

1. Stop all replication tasks and schedules for the replication configurations that use the SQL Server source database that you want to upgrade.
2. Upgrade the SQL Server database server.

For more information, see the Microsoft SQL Server documentation.

3. In the Data Replication Console, open a replication configuration that uses the SQL Server source database and then save the configuration again.

   Repeat this step for each configuration that uses the SQL Server source database.

4. Restart all of the replication tasks and schedules.

Handling Applier Failures

The Applier might end abnormally after different types of failures, including database, network, and system shutdowns or failures.

After an Applier task ends with an error or in response to the **Abort Task** option, do not edit any of the options that affect the distribution of transactions across threads and do not edit the recovery table on the target. Also, do not change the table mappings if transaction records were committed to the target. If you comply with these requirements, the Applier can resume apply processing transparently, without data loss, after you restart it.

During the recovery cycle, each Applier thread must be able to process the transaction records that were not successfully processed during the failed apply cycle. If the distribution of transaction records across the threads changes, the target database might have data integrity issues related to duplicate or skipped records.

Do not edit the following options that can affect the distribution of Applier threads:

- The **Applier threads** field on the **Runtime Settings** tab > **General** view.
- The following Applier thread distribution parameters on the **Runtime Settings** tab > **Advanced Settings** view:
  - apply.subtasks_enabled
  - apply.distribute_by_obj_size_threshold
  - apply.distribute_by_obj_size
  - apply.global_transaction_support
  - apply.max_subtasks
- The **Distribute by tables / rows** option on the **Map Columns** tab or in the **Customize Settings for the Selected Tables** window

Important: If you must change one of these options after a failure, ensure that the Applier completes the recovery cycle and then use the **Stop Schedule** option to stop replication tasks. The Applier task completes normal processing and then stops in a controlled manner. In this case, the recovery table, IDR_RECOVERY, contains records for each thread.

Also, do not change table mappings after the Applier ends abnormally if at least one Applier thread performed a commit on the target. The recovery table IDR_RECOVERY on the target database records the commits that the threads applied. If the table is empty, none of the threads committed records and you can change table mappings.

APPENDIX A

Troubleshooting

This appendix includes the following topics:

Collecting Diagnostic Data for Troubleshooting

You can use Data Replication to collect diagnostic data for troubleshooting the Data Replication Console and Data Replication tasks. Typically, you collect diagnostic data at the request of Informatica Global Customer Support.

Data Replication can collect the following information:

- Data Replication SQLite database files
- Task execution logs
- Server Manager logs
- Intermediate files
- Data Replication Console logs

Note: You can choose which files to include in the diagnostic data collection based on various criteria, such as the configuration and specified time period.

To collect diagnostic data about Data Replication tasks, Data Replication creates a Collect Diagnostic Data task and starts the task with the data collection settings that you define in the Data Replication Console.

Important: Only the idradmin user can create Collect Diagnostic Data tasks. However, if you do not delete a Collect Diagnostic Data task after collecting diagnostic data, any user can start the task again and collect the same set of diagnostic data.

After Data Replication collects the requested diagnostic data, the Server Manager Main server generates the diag.zip file in the *DataReplication_installation*/logs/*date*/*run_id* subdirectory.

Restriction: The Collect Diagnostic Data task cannot collect a requested file from a Data Replication installation on Windows if the file path is longer than the Windows limit of 260 characters.

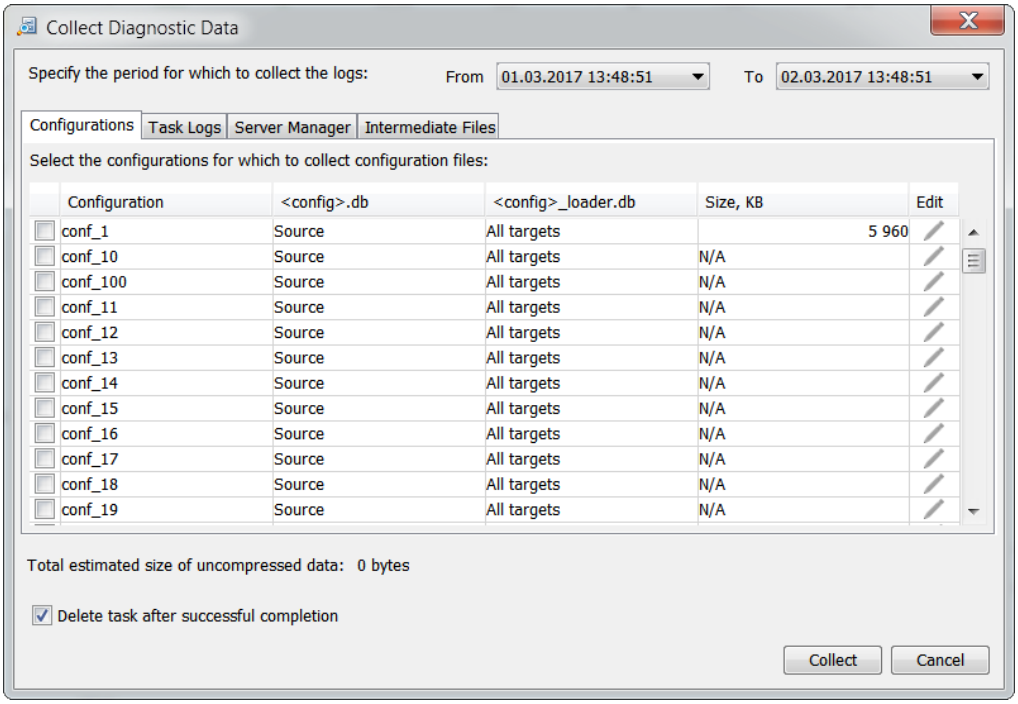
To collect diagnostic data about the local Data Replication Console installation, you do not have to connect to a Server Manager. Any user can create a zip file with the Data Replication Console log files.

Collecting Diagnostic Data for Troubleshooting Data Replication Tasks

From the Data Replication Console, the idradmin user can collect diagnostic data for replication tasks. Usually, this information is collected at the request of Informatica Global Customer Support.

1. In the Data Replication Console, connect to the Server Manager Main server as the idradmin user.
2. Click **Tools** > **Collect Diagnostic Data**.

   The **Collect Diagnostic Data** dialog box appears.



3. On the **Configurations** tab, select the check box next to each configuration that you want to include in the diagnostic data collection.

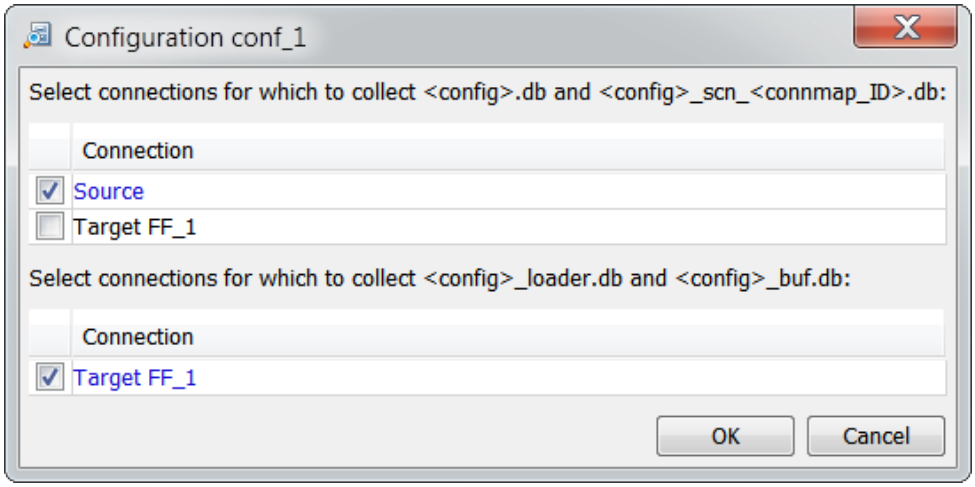
   By default, for each selected configuration, Data Replication will include the following SQLite database files:

   - The *config_name*.db and *config_name*_scn_*connection_object_ID*.db files that are associated with the source database connection
   - The *config_name*_loader.db and *config_name*_buf.db files from all of the target Server Manager instances for all target connections

4. Optionally, edit the list of SQLite configuration database files that you want to collect for each selected replication configuration. Perform the following substeps:

   a. Click the **Edit** icon in the selected configuration row.

      The **Configuration <configuration_name>** dialog box appears.

   b. Select the connections for which to collect the *config_name*.db, *config_name*_scn_*connection_object_ID*.db, *config_name*_loader.db and *config_name*_buf.db files.

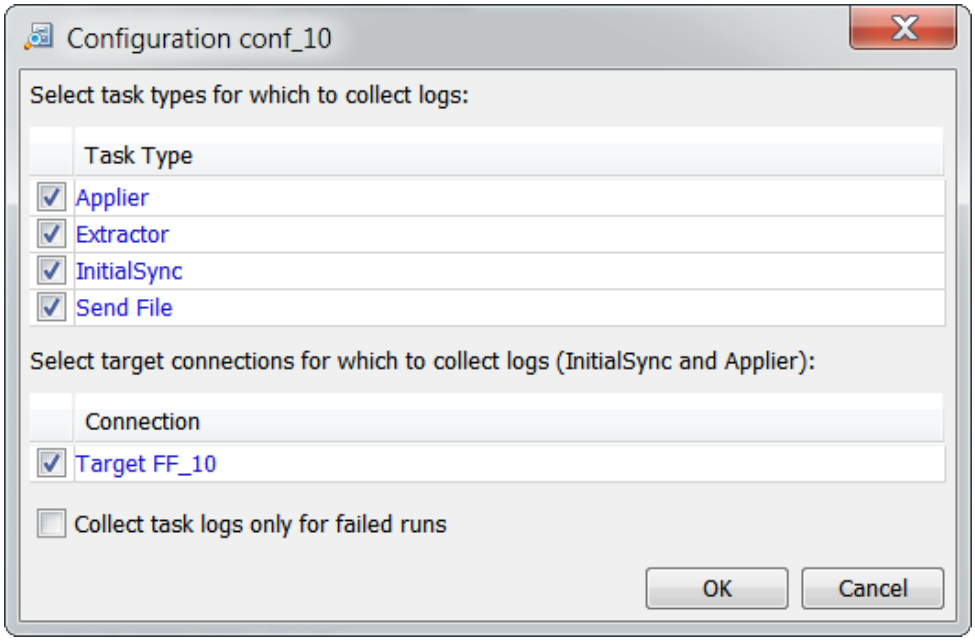
   c. Click **OK**.

5. On the **Task Logs** tab, select the check box next to each configuration for which you want to include execution logs in the diagnostic data collection.

   By default, for each selected configuration, Data Replication will include execution logs for all runs of all task types for the period of time that is defined by the **From** and **To** fields. InitialSync and Applier logs will be included for all targets.

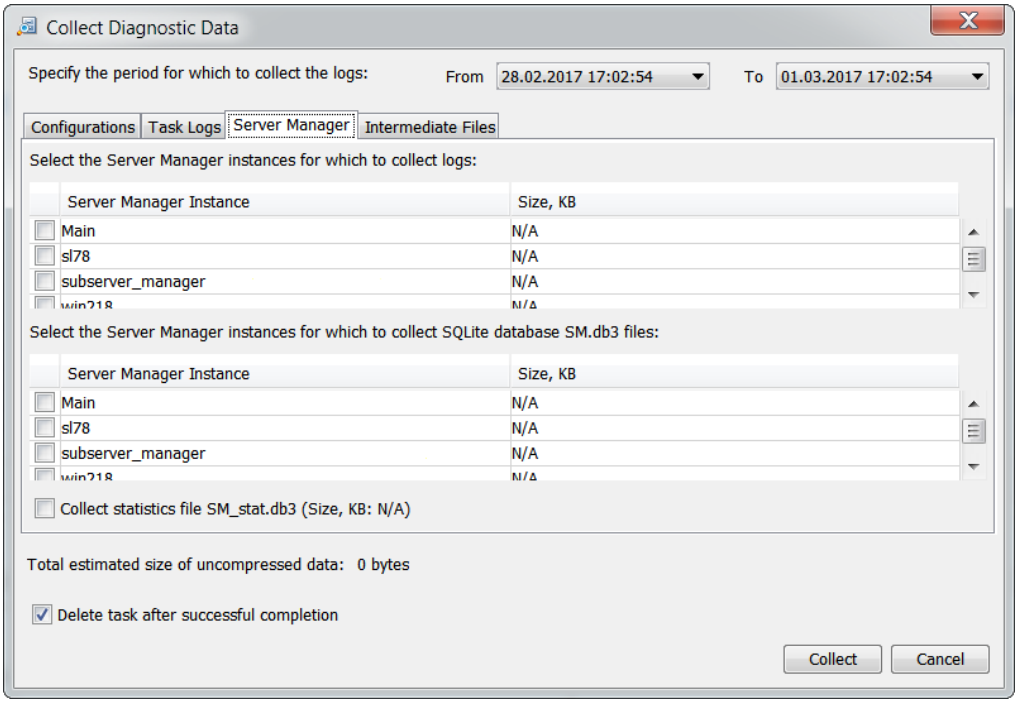
6. Optionally, edit the list of task types and runs for which you want to collect the log files. Perform the following substeps:
   a. Click the **Edit** icon in the selected configuration row.

      The **Configuration <configuration_name>** dialog box appears.

   b. Select the types of tasks for which you want to collect execution log files.
   c. Select the target connections for which you want to collect InitialSync and Applier logs.
   d. If you want to include log files only for the task runs that failed, select the **Collect task logs only for failed runs** option.
   e. Click **OK**.

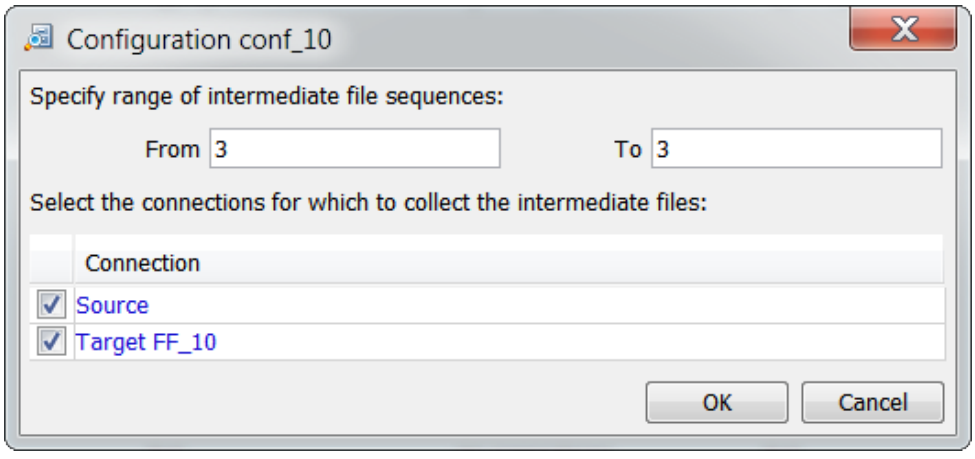
7. On the **Server Manager** tab, select the Server Manager instances for which you want to collect Server Manager log files and SQLite SM.db3 files.



8. To collect the SQLite database SM_stat.db3 file that contains task statistics, select **Collect statistics file SM_stat.db3**.
9. On the **Intermediate Files** tab, select the check box next to each configuration for which you want to include intermediate files in the diagnostic data collection.

   By default, Data Replication includes all of the intermediate files for all connections.
10. Optionally, edit the set of intermediate files that you want to include in the diagnostic data. Perform the following substeps:
    a. Click the **Edit** icon in the selected configuration row.

       The **Configuration <configuration_name>** dialog box appears.



    b. In the **From** and **To** fields, specify the range of intermediate files that you want to include.

c. In the list of available database connections, select source and target connections for which you want to collect intermediate files.

d. Click **OK**.

11. If you want to keep the Collect Diagnostic Data task after it collects the requested data, clear the **Delete task after successful completion** option.

    Important: If you keep the Collect Diagnostic Data task, any user can start this task again and collect the same set of diagnostic data. However, a regular user cannot change the data collection settings that were defined for this task.

12. Click **Collect**.

Data Replication creates a Collect Diagnostic Data task and starts the task with the specified data collection parameters. If the task ends successfully, it generates the diag.zip file that contains the collected diagnostic data. By default, Data Replication deletes the Collect Diagnostic Data task after it ends successfully.

Downloading the diag.zip File

After Data Replication collects the requested diagnostic data, the Server Manager Main server generates the diag.zip file that is located in the *DataReplication_installation*/logs/*<date>*/*<run_id>* subdirectory on the Server Manager Main server system. You can use the Data Replication Console to download the diag.zip file that contains the requested diagnostic data.

1. In the Data Replication Console, click the **Server Manager** tab > **Event Viewer** view.
2. Click the **Logs of Schedules** tab.
3. To filter the list of schedule instances and tasks, define any of the following filter criteria in the list boxes at the top:
   - In the **Start time** field and the **End time** field, enter a date and time to define the time period during which the schedule instances or tasks ran.
   - In the **Status** list, select one of the following schedule or task statuses: **Running**, **Success**, or **Failure**.

   Important: Do not change the default values of the **Schedule** and **Configuration** lists.

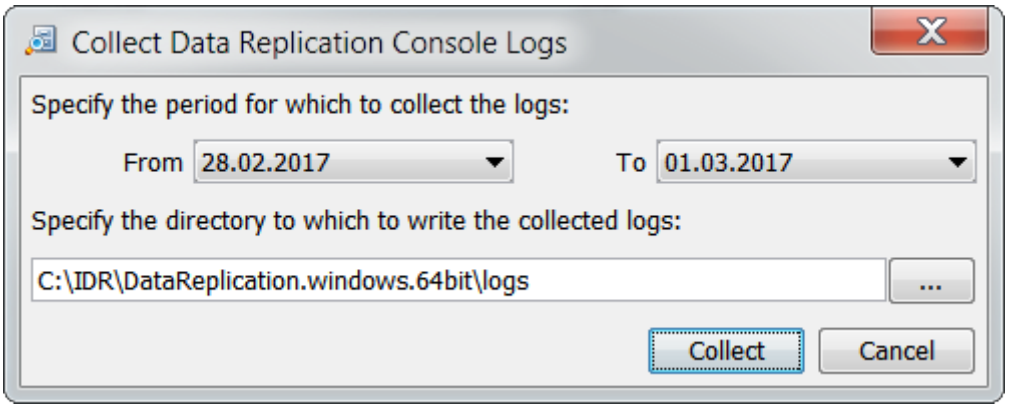
   Then click **Refresh** to display the filtered list, or select the **Autoupdate** check box to automatically refresh the filtered list every 5 seconds.
4. Select the row for the Collect Diagnostic Data task run in the upper list box.
5. Select the row for the Collect Diagnostic Data task in the lower list box and click the **Show Log Files** icon button.
6. In the **Task Information** window, select the diag.zip file and click **Download**.
7. Enter a file name and path and click **Save**.

Collecting Diagnostic Data for Troubleshooting the Data Replication Console

From the Data Replication Console, you can collect diagnostic data about the Data Replication Console. Usually, you collect this information at the request of Informatica Global Customer Support. The diagnostic data includes the Console log files and a text file that contains the revision number of the Console.

1. In the Data Replication Console, click **Tools** > **Collect Data Replication Console Logs**.

   The **Collect Data Replication Console Logs** dialog box appears.



2. In the **From** and **To** fields, enter dates to define the period for which to collect the logs.
3. To specify the directory to which to save the zip file that contains the collected log files and the text file with the Data Replication Console revision number, click the **Browse** button and then browse to the directory.
4. Click **Collect**.

The Data Replication Console creates a zip file with the collected diagnostic data in the specified directory.

Common Replication Problems

If you encounter a problem while using Data Replication, review the following list of previously reported issues and their solutions before contacting Informatica Global Customer Support.

When the Applier uses multiple threads to apply change data to a MySQL target, it ends with message IDR-070217, which reports the MySQL error "Lock wait timeout exceeded; try restarting transaction."

The Applier might end with error message IDR-070217 when applying change data to a MySQL target. This problem occurs if the Applier uses multiple threads to apply change data to the same target table simultaneously, resulting in a next-key lock. In this case, the MySQL database returns the error "Lock wait timeout exceeded; try restarting transaction." To resolve this issue, in the Data Replication Console, set the **Distribute By** option to **Tables** on the **Map Columns** tab for each of the target tables. For more information, see "Handling Applier Failures" on page 374.

After Data Replication is upgraded to a later version, the Applier ends with message IDR-070217, which reports the DB2 error "Could not convert binary other than LO type" or "Value too long for type character varying."

After you upgrade Data Replication to a later version, the Applier might end with error message IDR-070217 when processing data from a DB2 source. This problem occurs if the DB2 source database returns the error "Could not convert binary other than LO type" or "Value too long for type character varying." To resolve this issue, in the Data Replication Console, open a replication configuration that uses the DB2 source database and save the configuration again. Then restart all of the replication tasks and schedules.

An ORA-942 "table not found" error occurs when I select an Oracle table owner.

The connected user lacks some required user permission. For Oracle targets, see "Configuring Database User Privileges for Oracle Targets" on page 103. For Oracle sources, see "Configuring User Privileges for Oracle Sources" on page 79.

When I try to connect to a MySQL database, the following error occurs:

```
Could not connect to target: Communication failure during handshake. Is there server
running on localhost: 3306
```

If the MySQL process runs on the specified port but still receives the error, the issue probably occurs because you are using MySQL 4.1 or later.

During Data Replication Console startup, the following error occurs:

```
"'java' is not recognized as an internal or external command, operable program or batch
file."
```

The Data Replication Console is a Java application that requires the Java Runtime Environment (JRE) to be installed. Data Replication supports 64-bit JRE 1.7 and 1.8. If you determine that a supported JRE version is installed, the issue might be occurring because the java.exe or java executable is not specified in the PATH environment variable.

The Data Replication Console fails to start but does not display an error message.

To resolve the problem, perform one or more of the following actions:

- View the *DataReplication_installation*/logs/se.log file on the system where you run the Data Replication Console. The file might contain a message about the error that caused the Data Replication Console failure. Then try to resolve the problem and run the Data Replication Console again.
- If the *DataReplication_installation*/logs/se.log file is empty, on Windows, you can complete the following steps to try to display a message about the error in the Console:
  1. Create a backup copy of the idr_console.cmd script.
  2. Open the idr_console.cmd script in a text editor.
  3. In the following command, replace `start javaw` with `java`:

     ```
     start javaw -Dfile.encoding=utf-8 -Xms256m -Xmx1024m ^
     ```

     The resulting statement should appear as follows:

     ```
     java -Dfile.encoding=utf-8 -Xms256m -Xmx1024m ^
     ```
  4. Save and close the idr_console.cmd script.
  5. Run the Data Replication Console.
  6. Look for an error message in the Console window. Then based on the error information, try to resolve the problem.
  7. Replace the idr_console.cmd script with the backup copy and run the Data Replication Console again.

- If you have an old version of Java Runtime Environment (JRE), some java components might block the Data Replication Console. To resolve the issue, update JRE to the latest supported version.

If you still cannot start the Data Replication Console, contact Informatica Global Customer Support.

When I try to connect to Oracle while running the Data Replication engine, I get an error. However, when I connect to Oracle from the Data Replication Console, I do not get an error.

The following error is issued when I try to connect to Oracle while running the Data Replication engine:

```
TNS:listener could not resolve SERVICE_NAME given in connect descriptor.
```

The Data Replication Console uses a JDBC connection driver. With this driver, the Data Replication Console first tries to use the Oracle SID and then the SERVICE_NAME to establish a connection. The Data Replication engine uses the SERVICE_NAME by default to connect to the database.

The SERVICE_NAME is specified in a tnsnames.ora connection string as follows:

```
(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=hostname)(PORT=port)))
(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=instancename)))
```

To resolve this issue, try using the Oracle SID value. Either select the **Use SID instead of SERVICE_NAME** option in the Data Replication Console or specify the following parameter in the configuration .xml file:

```
<UseSidInsteadOfServiceName>true</UseSidInsteadOfServiceName>
```

When the Data Replication Extractor runs on AIX, the following error occurs:

```
ACE_Message_Block[14:43:55.719] Processing file:/home/user/IDR/AIX/DBSync/cust_log/
log_1098 status:0

[14:43:55.732] ACE_OS::pread() read -1 bytes. Error code: 14. Address: 0
fffffffffff0cc8. Block size: 512. Offset: 512.

[14:43:55.737] OracleLogParser::parseSingleLog parsing failed for /home/user/IDR/AIX/
DBSync/cust_log/log_1098
```

If this error occurs on AIX or another Linux or UNIX operating system, try increasing the operating system ulimit value for the data area size to a value such as 170000. For more information about the ulimit command, see your operating system documentation.

The Server Manager and replication tasks use excessive amounts of virtual memory

The Server Manager or replication tasks that run on Linux with the glibc library version 2.10 or later might use excessive amounts of virtual memory.

This problem might be related to the enhanced dynamic memory allocation (malloc) behavior of the glibc library 2.10 or later.

To reduce virtual memory usage, set the MALLOC_ARENA_MAX environment variable to 4. Then restart the Server Manager that runs on this computer.

To reduce virtual memory usage of a Server Manager instance, you can also reduce the number of threads that the Server Manager uses. Adjust the following Server Manager properties:

- ApiThreadsCount
- BuiltinTaskManagersCountThreads
- CountThreads
- IntermediateFileTransfersCountThreads
- SendCountThreads
- TransferThreadPoolsCountThreads

You can also set the DisableStatistics advanced property for the Server Manager to 1 to stop the collection of latency statistics.

I receive one of the following warning or error messages that indicates that the available disk space for the Server Manager is low or critically low.

When trying to connect to the Server Manager, I receive the following error message:

```
Not enough disk space on the Server Manager file system.
```

I receive one of the following error messages in the Event Viewer or in a notification email:

```
[WARNING] Disk space on the system of the Server Manager server_name is low. Available
disk space for the following directories is less than the low disk space threshold of
disk_space GB that is specified in the WarningValueFreeSpace parameter:
list_of_directories.
```

```
[ERROR] Disk space on the system of the Server Manager server_name is critically low.
The Server Manager stopped all of the replication tasks. Available disk space for the
following directories is less than the minimum disk space of required_space GB that is
specified in the LimitValueFreeSpace parameter: list_of_directories.
```

To avoid the disk space error messages, provide sufficient memory resources for the specified Server Manager instance. To avoid the disk space warning message, you can also decrease the value of the WarningValueFreeSpace parameter. You can edit the Server Manager parameters on the **Advanced Settings** tab in the Server Manager **Properties** dialog box. For more information about setting advanced properties for the Server Manager, see "Editing Properties for the Main Server or a Subserver" on page 138.

Tip: You can view the available disk space for the directories that the Server Manager uses on the **Version** tab in the Server Manager **Properties** dialog box. For more information about viewing details for the Server Manager, see "Viewing Information About the Server Manager System" on page 154.

After an Applier task ends with an error or in response to the **Abort Task** option, the following Teradata error occurs during subsequent Applier runs:

```
error 2652: Operation not allowed: schema_name.table_name is being Loaded.
```

If the Applier task ends abnormally or is aborted when loading source data to audit log tables on a Teradata target, the task might lock the audit log tables. If the audit log tables are locked, during subsequent Applier runs, the Applier task ends with an error. To continue replication, resolve any problem that led to the abnormal completion of the task. Then, drop the locked audit log tables, re-create the audit log tables on the Teradata target, and restart the Applier task.

When trying to replicate source data to an Apache Kafka target, the Applier ends with message IDR-12004, which reports the error "Could not create the Java Virtual Machine."

The Applier spawns a Java Virtual Machine (JVM) to execute the code required to apply data to Kafka targets. When this error occurs, it indicates that the Server Manager PATH or LIBRARY_PATH path environment variable specifies a directory that does not contain the necessary Java JVM library.

Edit the appropriate environment variable on the system on which the Applier runs to specify a directory that includes the Java JVM library:

- On Windows, add the directory that contains the jvm.dll library to the PATH environment variable. For example, use the following command:

  ```
  PATH=%PATH%;%JAVA_HOME%\jre\bin\server
  ```

- On Linux, add the directory that contains the libjvm.so library to the LD_LIBRARY_PATH path environment variable. For example, use the following command:

  ```
  LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/amd64/server
  ```

Then stop and restart the Server Manager. The JVM library directory must be specified in the PATH or LIBRARY PATH of your environment before you start the Server Manager on the target system.

When trying to replicate source data to an Apache Kafka target, I receive a Java Virtual Machine OutOfMemory error.

The Applier spawns a Java Virtual Machine (JVM) to execute the code required to apply data to Kafka targets. When a JVM OutOfMemory error occurs, it indicates that the JVM is attempting to consume more memory than the maximum it has been configured to consume. You can specify the maximum memory setting assigned to the JVM by changing the value of the LIBHDFS_OPTS environment variable. The Applier passes the value specified for the LIBHDFS_OPTS environment variable to the JVM.

For example, use the following command to set the maximum heap memory size to 1000 megabytes:

```
LIBHDFS_OPTS=-Xmx1000m
```

Notes:

- Set the LIBHDFS_OPTS environment variable before starting the Server Manager for the Applier.
- The Applier does not explicitly set any memory limits or defaults for the JVM. The JVM will use its own default settings, which vary by JVM vendor and operating environment.
- You can use the LIBHDFS_OPTS environment variable to set any property that the JVM accepts. However, setting memory limits is the most common use for the LIBHDFS_OPTS environment variable.

When trying to replicate source data to an Apache Kafka target, the Applier ends with message IDR-130003 QADPTR-15028, which reports the following error:

```
An attempt to connect to Kafka was unsuccessful because of a Class Not Found exception.
A third-party library is probably missing from the generated Java classpath.
```

Typically, this message indicates that not all Kafka client library files have been added to the generated classpath. Verify that the apply.kafka.kafka_client_libraries_directory runtime parameter specifies the correct location of Kafka client library files on the machine where the Applier runs and that the version of the client library files matches the Kafka target version. The Kafka client libraries are JAR files located in the Kafka installation libs directory. For example, *kafka_2.11-0.10.0.1*/libs. For more information about the apply.kafka.kafka_client_libraries_directory runtime parameter, see Appendix C, “Data Replication Runtime Parameters” on page 396.

For Oracle targets, the Applier and InitialSync use excessive amounts of memory

The Applier might use an excessive amount of memory when applying change data to LOB columns in Oracle target tables in either Audit Apply or Merge Apply mode. Also, InitialSync might use an excessive amount of memory when loading data to Oracle LOB columns. To reduce the memory usage, try upgrading the Oracle Client on the systems where the Applier and InitialSync run to the Client for the latest Oracle version that Data Replication supports.

InitialSyncs ends with error message IDR-000413, which reports the following DB2 for Linux, UNIX, and Windows driver error for a DB2 target:

```
[IBM][CLI Driver][DB2/NT64] SQL0964C The transaction log for the database is full.
SQLSTATE=57011.
```

InitialSync might end with this error because the DB2 for Linux, UNIX, and Windows target database is not configured to handle the amount of data that InitialSync attempts to commit to the target. To resolve this issue, perform one of the following steps:

- In the Data Replication Console, decrease the value of the initial.rows_to_commit runtime parameter. This parameter specifies the maximum number of Insert rows that a single InitialSync thread can commit on the target.

- Increase the size of the DB2 transaction log.

APPENDIX B

Data Replication Files and Subdirectories

This appendix includes the following topics:

Files and Subdirectories

The Informatica Data Replication installation root directory contains subdirectories, executables, scripts, and other files. Read the descriptions of these files and subdirectories to determine which one you need to use.

Data Replication Script Files

Script files end in `*.sh` or `*.cmd`. These files start Informatica Data Replication components and modules. Files that have the extension .sh run on Linux or UNIX, and files that have the extension .cmd run on Windows. These shell scripts set the appropriate environment variables, such as LD_LIBRARY_PATH, before running the executable.

Data Replication provides the following script files in the root installation directory:

Informatica Data Replication Console

idr_cmd_line.sh or idr_cmd_line.cmd

Starts the Replication Configuration Command Line Interface.

idr_console.cmd

Starts the Data Replication Console. The Data Replication Console is supported only on Windows.

InitialSync

initialsync.sh or initialsync.cmd

When either the source or target is Oracle, synchronizes the source and target tables.

initialsync_odbc.sh or initialsync_odbc.cmd

When neither the source nor target is Oracle, synchronizes the source and target tables.

initialsync_teradata.sh or initialsync_td.cmd

When the source is Oracle and the target is Teradata, synchronizes the source and target tables.

initialsync_odbc_teradata.sh or initialsync_odbc_td.cmd

When the source is a database other than Oracle and the target is Teradata, synchronizes the source and target tables.

initialsync_direct.sh

When the target is Oracle, synchronizes the source and target tables using a DIRECT command line parameter setting of **y**.

Extractor

extract.sh or extract.cmd

Starts the Informatica Data Replication extraction engine based on the configuration file that you generated from the Data Replication Console.

extract_db2.sh or extract_db2.cmd

When the source database is DB2 for Linux, UNIX, and Windows, starts the Informatica Data Replication extraction engine based on the configuration file.

extract_mssql.sh or extract_mssql.cmd

When the source database is Microsoft SQL Server, starts the Informatica Data Replication extraction engine based on the configuration file.

Applier

apply.sh or apply.cmd

Starts the Informatica Data Replication apply engine based on the configuration file that you generated from the Data Replication Console.

apply_db2.sh or apply_db2.cmd

For DB2 targets, starts the Data Replication apply engine to apply captured changes to the target database.

apply_oracle.sh or apply_oracle.cmd

For Oracle targets, starts the Data Replication apply engine to apply captured changes to the target database.

apply_odbc.sh or apply_odbc.cmd

For ODBC-enabled targets, starts the Data Replication apply engine to apply captured changes to the target databases.

apply_datadirect.sh

For targets with DataDirect drivers, starts the Data Replication apply engine to apply captured changes to the target databases.

apply_td.sh or apply_teradata.cmd

For Teradata targets, starts the Data Replication apply engine to apply captured changes to the target databases.

Server Manager

idrcmd.sh or idrcmd.bat

Starts the Server Manager Command Line Interface.

install_sm_service.cmd

Installs the Data Replication Server Manager as a service on Windows.

server_manager.sh or server_manager.cmd

Starts the Data Replication Server Manager.

start_sm_service.cmd

Starts the Data Replication Server Manager as a service on Windows.

stop_sm_service.cmd

Stops a Data Replication Server Manager service that is running on Windows.

uninstall_sm_service.cmd

Uninstalls a Data Replication Server Manager service on Windows.

Executables Called from the Data Replication Console or Scripts

The following table identifies, by target type, the InitialSync and Applier executables that can be called from the Data Replication Console or scripts:

Target Types	InitialSync Executables	Applier Executables
Amazon Redshift	- dbsync_initial_redshift.exe. Use this executable on Windows to synchronize Oracle sources with Amazon Redshift targets. - dbsync_initial_odbc_redshift.exe. Use this executable on Windows to synchronize sources other than Oracle with Amazon Redshift targets. - dbsync_initial_redshift. Use this executable on Linux and UNIX to synchronize Oracle sources with Amazon Redshift targets. - dbsync_initial_odbc_redshift. Use this executable on Linux and UNIX to synchronize sources other than Oracle with Amazon Redshift targets.	- dbsync_apply_redshift.exe. Use on Windows. - dbsync_apply_redshift. Use on Linux and UNIX.
DB2 for Linux, UNIX, and Windows	- dbsync_initial.exe. Use this executable on Windows to synchronize Oracle sources with DB2 targets. - dbsync_initial_odbc.exe. Use this executable on Windows to synchronize sources other than Oracle with DB2 targets. - dbsync_initialsync. Use this executable on Linux and UNIX to synchronize Oracle sources with DB2 targets. - dbsync_initial_odbc. Use this executable on Linux and UNIX to synchronize sources other than Oracle with DB2 targets.	- dbsync_apply_db2.exe. Use on Windows. - dbsync_apply_db2. Use on Linux and UNIX.
Oracle	- dbsync_initial.exe. Use on Windows. - dbsync_initialsync. Use on Linux and UNIX.	- dbsync_apply_oracle.exe. Use on Windows. - dbsync_apply_oracle. Use on Linux and UNIX.
Teradata	- dbsync_initial_td.exe. Use this executable on Windows to synchronize Oracle sources with Teradata targets. - dbsync_initial_odbc_td.exe. Use this executable on Windows to synchronize sources other than Oracle with Teradata targets. - dbsync_initial_td. Use this executable on Linux and UNIX to synchronize Oracle sources with Teradata targets. - dbsync_initial_odbc_td. Use this executable on Linux and UNIX to synchronize sources other than Oracle with Teradata targets.	- dbsync_apply_td.exe. Use on Windows. - dbsync_apply_td. Use on Linux and UNIX.

Target Types	InitialSync Executables	Applier Executables
- Greenplum - Microsoft SQL Server - MySQL - Netezza - PostgreSQL - Vertica	- dbsync_initial.exe. Use this executable on Windows to synchronize Oracle sources with the specified targets. - dbsync_initial_odbc.exe. Use this executable on Windows to synchronize sources other than Oracle with the specified targets. - dbsync_initialsync. Use this executable on Linux and UNIX to synchronize Oracle sources with the specified targets. - dbsync_initial_odbc. Use this executable on Linux and UNIX to synchronize sources other than Oracle with the specified targets.	- dbsync_apply_odbc.exe. Use on Windows. - dbsync_apply_odbc. Use on Linux and UNIX.
- Apache Kafka - Cloudera - Hortonworks - Flat File	- Not applicable	- dbsync_apply_odbc.exe. Use on Windows. - dbsync_apply_odbc. Use on Linux and UNIX.

The following table identifies, by source type, the Extractor executables:

Source	Extractor
DB2 for Linux, UNIX, and Windows	- dbsync_db2_extract.exe. Use on Windows. - dbsync_extract_db2. Use on Linux and UNIX.
Microsoft SQL Server	- dbsync_mssql_extract.exe. Use on Windows. - dbsync_mssql_extract. Use on Linux and UNIX.
MySQL	- dbsync_mysql_extract.exe. Use on Windows. - dbsync_mysql_extract. Use on Linux.
Oracle	- dbsync_extract.exe. Use on Windows. - dbsync_extract. Use on Linux and UNIX.

Related Topics:

- "About Command Line Parameters" on page 438

Default.cfg File

Data Replication versions earlier than 9.7.0 supplied the default.cfg file in the *DataReplication_installation*/uiconf directory so that users could override the default configuration values for the Data Replication Console. Beginning with Data Replication 9.7.0, the default.cfg file is no longer installed because overriding the defaults should be done with caution and only at the direction of Informatica Global Customer Support or a product specialist.

If an Informatica Global Customer Support or a product specialist directs you to add or change a parameter value, you can edit the default.cfg file if it exists or use a text editor to create the file in the uiconf directory. You can then enter parameter changes.

To add or edit a parameter value, perform the following steps:

1. Open the default.cfg file in a text editor.
2. Enter the parameter in the following format:

   ```
   parameter_name=parameter_value
   ```

   An Informatica representative will provide the parameter name and value. If you need to specify multiple parameters, enter each parameter on a separate line.
3. Save the file.

For example, to disable mapping validation in the Data Replication Console, change the compatibility_checking_level parameter from 4 (the default) to 0 by adding the following line to the default.cfg file:

```
compatibility_checking_level=0
```

Related Topics:

- "Start Point for the Extractor Task" on page 30
- " Coordination with Microsoft SQL Server Change Data Capture" on page 70
- "Schema Generation Parameters" on page 201
- "Mapping Source and Target Tables" on page 209
- "Mapping Validation" on page 218
- "Filtering Table Rows for Selective Replication to Different Targets" on page 301
- "Managing Open Transactions" on page 336
- "Updating Source and Target Metadata for a Configuration in Non-interactive Mode" on page 453

Other Key Files

Data Replication provides the following additional key files in the installation root directory:

DataReplication.key

Contains the Data Replication license key value.

config.xsd

An xml schema file that defines the format of the Data Replication configuration files.

SM.db3, SM.db3-shm, SM.db3-wal

The SQLite database files that store the Server Manager settings.

SM_stat.db3

The SQLite database that stores the statistical data for the Server Manager. The Server Manager uses it to calculate the latency of the end-to-end replication.

Subdirectories

A Data Replication installation includes the following subdirectories that appear after installation or that are generated subsequently:

Backup_*timestamp*

Initially, this subdirectory does not exist. If you delete the Server Manager SQLite database, sm.db3, in the Data Replication installation directory and then start the Server Manager, it moves the internal master key file, configuration SQLite databases, and intermediate files that exist at the time of the Server Manager restart to this directory.

checkpoints

Initially, this subdirectory does not exist. If you have one or more configurations with an Apache Kafka target, Data Replication automatically creates a checkpoint file in the checkpoints subdirectory when you apply data to a Kafka target. By default, the name of each checkpoint file matches the configuration name. You can change the checkpoint file name and directory by editing the apply.kafka.kafka_file_checkpoint_file_name and apply.kafka.kafka_checkpoint_file_directory parameters on the **Runtime Settings** tab > **Advanced Settings** view of the Data Replication Console.

configs

Contains Data Replication configuration files and the Extractor and Applier SQLite databases. For each configuration file, Data Replication generates the following directory and files:

- *<config_name>_<target_number>*. A subdirectory for a configuration that contains the SQLite database files that store the internal processing information that the Applier task added for each target. When a replication has multiple targets, Data Replication generates a separate subdirectory for each target.
- *<config_name>*.db, *<config_name>*.db-journal or *<config_name>*.db-wal, *<config_name>*.db-shm. The SQLite database files that store the configuration file and the internal processing information that the Extractor task added.

Do not edit the files in this directory. They are for internal Data Replication use only.

dd/lib

On Linux and UNIX, contains the DataDirect ODBC driver libraries.

doc

Contains Data Replication user manuals, including this user guide. The **samples** > **deploy** subdirectory contains a sample deploy.properties file for deploying a configuration in the Server Manager Command Line Interface. Specify the settings for the configuration you want to deploy, and then save the file with another file name in any directory.

dump

Data Replication generates files in this directory to store invalid records that the Oracle Extractor cannot parse.

lib

On Windows, contains java libraries for the Data Replication Console, including JDBC drivers to connect to various databases. The infaLib subdirectory contains .jar libraries that are loaded dynamically at runtime.

lib/hadoop

Initially, this subdirectory does not exist. When you extract the hadoop_libs.zip package into the *DataReplication_installation* directory, this subdirectory is created and contains the .jar files that are required for Cloudera and Hortonworks targets.

lib/queueAdapterLib

For Apache Kafka targets, this subdirectory is the default location for the Kafka client library files. Ensure that the subdirectory exists on the machine where the Data Replication Applier runs. Copy the Kafka library files from the *kafka_version*/libs directory to this subdirectory. The version of the library

files must match the version of Kafka to which Data Replication connects. To override this subdirectory, use the apply.kafka.kafka_client_libraries_directory runtime parameter.

logs

Initially, this subdirectory is empty. When you run the Data Replication Console, Server Manager, and replication tasks, Data Replication generates the following types of log files in this directory or in a subdirectory within it:

- The Data Replication Console log file, se.log, in the UI subdirectory. This log file contains error and debug information for a Console session.
- A deployment log file, deployment.log, in the UI subdirectory. This log file contains error, status, and warning messages that the Console displays during local or remote deployment of a configuration or configuration changes.
- Server Manager log files in the SM subdirectory. These log files contain error and debug information for a Server Manager execution.
- Log files for replication tasks, including Extractor, Applier, and Send File tasks. The Server Manager groups these logs by execution date and stores them in the appropriate subdirectories.

messages

Contains the error message files that the Data Replication Console displays for Data Replication tasks and processes. The subdirectory also contains separate message files for replication tasks performed with a specific source or target database.

overflow

For Apache Kafka targets, contains the Least Recently Used (LRU) cache overflow files from the Queue Adapter and an Avro schema cache file for each mapped source table. The LRU cache files contain definitions of the structure of the source input. The Avro schema cache files contain definitions of the structure of the output messages. The Avro schema cache file names have the following format: key*key_number_source_owner_source_table_name*.che. Initially, this subdirectory does not exist. When the Applier first runs, this subdirectory and its subdirectories and files are created dynamically. The LRU cache overflow files are typically cleared after each Applier run, whereas the Avro schema cache files are persisted across Applier runs.

output*configuration_name*

Contains the intermediate files that Data Replication generates for a configuration when a replication job runs. Each configuration has a separate *configuration_name* subdirectory. The subdirectory contains both the data and transaction files that comprise the intermediate files. The data files are .dat files. The transaction files are .trn files.

Also, this subdirectory can contain a tmp subdirectory for the temporary spill files that the Applier creates when the memory buffers that accumulate data for long-running transactions and target tables exceed the size limit for these buffers.

scripts

On Linux and UNIX, contains the script that sets the environment variables to configure the DataDirect ODBC drivers. Data Replication script files call the script prior to running the replication tasks.

support

Contains runtime libraries that the Data Replication executables use. The oci_stub subdirectory contains the Oracle Client library stub, which is required for all sources and targets.

uiconf

Contains global configuration files that are used and updated by the Data Replication Console.

This subdirectory also contains the default.cfg file if you edited any of the default configuration parameters in a Data Replication installation before version 9.7.0, or if you create the file to modify specific default configuration parameters at the direction of Informatica Global Customer Support. For more information about the default.cfg file, see “Default.cfg File” on page 391.

APPENDIX C

Data Replication Runtime Parameters

Depending on the source and target types that you use for a replication job, the Data Replication Console lists the runtime parameters that are available for the Extractor, Applier, and InitialSync components.

In the Data Replication Console, you can optionally customize runtime parameters on the **Runtime Settings** tab > **Advanced Settings** view. The following list describes the runtime parameters that you can edit in the Console.

Amazon Redshift Parameters

redshift.s3.bucket_name

Specifies the name of the Amazon S3 bucket that stores the temporary files that contain the data to be applied to the Amazon Redshift target.

redshift.s3.file_size

Specifies the maximum size of the temporary files, in kilobytes, that Data Replication creates in the Amazon S3 bucket before loading the source data to Amazon Redshift.

Note: Data Replication treats this parameter value as a soft limit on the temporary file size.

Valid values are 1700 to 100000.

Default value: 5120 KB

redshift.s3.key_id

Specifies an AWS access key ID that Data Replication must use to access the Amazon S3 account resources including the bucket where Data Replication creates temporary files.

redshift.s3.path

Specifies the name of the directory in the Amazon S3 bucket where Data Replication creates the temporary files that store source data.

redshift.s3.secret_key

Specifies the secret access key for the access key ID that is specified in the redshift.s3.key_id parameter. The access key ID must have the authority to access the Amazon S3 account resources.

Apache Kafka Parameters

apply.avro.avro_before_image_prefix

Specifies a prefix for the names of Avro message fields that contain the before image data for an Update operation. Valid values are the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.

You can also specify a suffix in the apply.avro.avro_before_image_suffix parameter. If you specify both a prefix and suffix, the Avro before-image field name uses the following format: *<prefix><source_column_name><suffix>*.

Note: If you do not specify a value for this parameter or the apply.avro.avro_before_image_suffix parameter, the Avro before image field name duplicates the source column name. Informatica recommends that you enter a value in at least one of these parameters to create unique before-image field names.

For this parameter to be used, the apply.avro.avro_include_before_image parameter must be set to 1.

Default value: Null

apply.avro.avro_before_image_suffix

Specifies a suffix for the names of the Avro message fields that contain the before image data for an Update operation. Valid values are the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.

You can also specify a prefix in the apply.avro.avro_before_image_prefix parameter. If you specify both a suffix and prefix, the Avro before-image field name uses the following format: *<prefix><source_column_name><suffix>*.

Note: If you do not specify a value for this parameter or the apply.avro.avro_before_image_prefix parameter, the Avro before-image field name duplicates the source column name. Informatica recommends that you enter a value in at least one of these parameters to create unique before-image field names.

For this parameter to be used, the apply.avro.avro_include_before_image parameter must be set to 1.

Default value: _OLD

apply.avro.avro_include_before_image

Indicates whether Data Replication creates Avro message fields that contain the before image of updated columns when replicating an Update operation to a Kafka target. Valid values are:

- **0**. Do not create fields that contain the before image of columns.
- **1**. Create fields that contain the before image of columns.

Default value: 1

To add a prefix or suffix to the names of before-image fields, use the apply.avro.avro_before_image_prefix and apply.avro.avro_before_image_suffix parameters.

apply.avro.avro_include_is_present

Indicates whether Data Replication creates Avro message fields that contain a Boolean value indicating one of the following reasons for the occurrence of a null value in an Avro message field:

- The corresponding source column value is also null (null "is present").
- The corresponding source column has a value that is not null (null is not present). However, Data Replication does not capture the value because it did not change during the current replication cycle.

Valid values are:

- **0**. Do not create fields that contain a Boolean value indicating whether the null value is present or is not present in the source data.
- **1**. Create fields that contain a Boolean value indicating whether the null value is present or not present in the source data.

Default value: 1

To add a prefix or suffix to the names of these fields, use the apply.avro.avro_is_present_prefix and apply.avro.avro_is_present_suffix parameters.

apply.avro.avro_is_present_prefix

Specifies a prefix for the named of the Avro message fields that indicate whether a null value in an Avro message is also present in the corresponding source data. Valid values are the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.

You can also specify a suffix in the apply.avro.avro_is_present_suffix parameter. If you specify both a prefix and suffix, the Avro field name uses the following format: *<prefix><source_column_name><suffix>*.

Note: If you do not specify a value for this parameter or the apply.avro.avro_is_present_suffix parameter, the names of the Avro fields that indicate whether a null is present in the source data duplicate the source column names. Informatica recommends that you enter a value in at least one of these parameters to create unique field names.

For this parameter to be used, the apply.avro.avro_include_is_present parameter must be set to 1.

Default value: Null

apply.avro.avro_is_present_suffix

Specifies a suffix for the names of the Avro message fields that indicate whether a null value in an Avro message is also present in the corresponding source data. Valid values are the digits 0-9, Latin letters A-Z and a-z, and the underscore (_) character.

You can also specify a prefix in the apply.avro.avro_is_present_prefix parameter. If you specify both a suffix and prefix, the field name uses the following format: *<prefix><source_column_name><suffix>*.

Note: If you do not specify a value for this parameter or the apply.avro.avro_is_present_prefix parameter, the names of the Avro fields that indicate whether a null is present in the source data duplicates the source column names. Informatica recommends that you enter a value in at least one of these parameters to create unique field names.

For this parameter to be used, the apply.avro.avro_include_is_present parameter must be set to 1.

Default value: _PRESENT

apply.avro.avro_metadata_prefix

Specifies a prefix for the names of any metadata fields that are added to an Avro schema to distinguish these metadata fields from the source columns. You can also specify a suffix by using the apply.avro.avro_metadata_suffix parameter. If you specify both a prefix and suffix, the names of the metadata fields use the following format: *<prefix><metadata_field_name><suffix>*.

Default value: Null

apply.avro.avro_metadata_suffix

Specifies a suffix for the names of any metadata fields that are added to an Avro schema to distinguish these metadata fields from the source columns. You can also specify a prefix by using the apply.avro.avro_metadata_prefix parameter. If you specify both a suffix and prefix, the names of the metadata fields use the following format: *<prefix><metadata_field_name><suffix>*.

Default value: Null

apply.avro.avro_name_replacement_value

Specifies a character that will replace any unsupported character in source column or table names when Data Replication generates Avro field names for Kafka messages. Avro field names do not allow spaces or special characters. If this parameter is not specified, Data Replication will not replace unsupported Avro characters from source column or table names. For example, if this parameter specifies the underscore (_) character and a source column is named `column#name with space`, the corresponding Avro field name is `column_name_with_space`.

Default value: Underscore (_)

apply.avro.avro_schema_cache_directory

Specifies the directory where the Avro formatting code stores Avro schema objects that are used to produce Avro messages for each mapped source table.

Default value: *DataReplication_installation*/overflow

apply.avro.avro_schema_cache_memory_size

Specifies the maximum number of Avro schema files in a single Least Recently Used (LRU) cache instance to keep in memory. Data Replication writes an Avro schema file for each mapped source table in a configuration to the directory specified in the apply.avro.avro_schema_cache_directory parameter. When the number of Avro schema files in the cache instance exceeds the limit specified in this parameter, the excess files are removed from memory.

Valid values are 10 through 1,000,000,000

Default value: 1000

apply.kafka.kafka_checkpoint_file_directory

Specifies the directory to which Data Replication writes checkpoint files when you apply data to a Kafka target. The directory must be on the system where the Applier runs.

Default value: *DataReplication_installation*/checkpoints

apply.kafka.kafka_file_checkpoint_file_name

Specifies the name of the checkpoint file that Data Replication creates when you apply data to a Kafka target. When the value of this parameter is DEFAULT, each checkpoint file name matches the corresponding configuration name.

Default value: DEFAULT

apply.kafka.kafka_client_libraries_directory

Specifies the directory that Data Replication uses to access Kafka client libraries. Copy the files from the *kafka_version*/libs directory to the default Data Replication directory or enter the path to another directory. The library files must match the version of Kafka to which Data Replication connects, and the files must be located in a directory that runs on the same machine as the Data Replication Applier.

Default value: *DataReplication_installation*/lib/queueAdapterLib.

apply.kafka.kafka_message_key

Specifies an optional message key that Data Replication uses to assign a topic partition to the Avro messages sent to a Kafka target. All messages that use the same topic and message key are assigned to the same partition in the topic. If you specify USE_TABLE_NAME, Data Replication uses the name of each mapped source table as the key name in the messages.

Default value: Null

apply.kafka.kafka_producer_guarantee_delivery

Specifies whether or not to enable guaranteed delivery of messages to a Kafka target. When guaranteed delivery is enabled, the Queue Adapter attempts to guarantee that no messages are lost or duplicated on the

target if a network outage occurs. Guaranteed delivery provides for the highest level of data integrity on the target but might degrade performance. Valid values are:

- **0**. Disable guaranteed delivery. The Applier writes a checkpoint to the checkpoint file after each Commit operation, or after the timeout period that is specified in the apply.qadapter.checkpoint_request_timeout runtime parameter elapses. If you restart apply processing after an outage, duplicate or missing messages might occur on the target.
- **1**. Enable guaranteed delivery. The Applier writes a checkpoint to the checkpoint file after each message is successfully sent to the target topic. All messages are sent synchronously if the apply.kafka.kafka_producer_send_synchronously runtime parameter is set to 1. If you restart apply processing after an outage, no duplicate or missing messages should occur on the target.

Default value: 1

apply.kafka.kafka_producer_max_retry_attempts

Specifies the maximum number of times that the Queue Adapter Connector will retry sending a Kafka message that produced an error.

Default value: 5

apply.kafka.kafka_producer_partition_ID

Specifies the topic partition ID that Data Replication will include in each message sent to Kafka. A value of -1 causes Data Replication to send a null partition ID to Kafka.

Default value: -1

apply.kafka.kafka_producer_send_synchronously

Controls whether to use synchronous delivery of messages to Kafka. When synchronous delivery is enabled, Kafka must acknowledge each message as received before Data Replication will send the next message. In this mode, Kafka is unlikely to receive duplicate messages, but apply performance might be slower. Valid values are:

- **0**. Disable synchronous delivery. The Applier sends messages asynchronously to the target.
- **1**. Enable synchronous delivery.

Default value: 0

Note: With either option, the Queue Adapter maintains changes in commit order as they are streamed to the target.

apply.kafka.kafka_table_name_include_schema

When USE_TABLE_NAME is specified in the Data Replication Console or in the apply.kafka.kafka_message_key parameter to use the mapped source table name as the Kafka topic name or message key name, you can use this parameter to also include the source table schema name in the topic and key names. Valid values are:

- **0**. Do not include the source table schema name in the topic and key names.
- **1**. Include the source table schema name in the topic and key names. With this setting, a Kafka topic name will have the following format: *schema_name*.*source_table_name*.

Default value: 0

apply.kafka.kafka_table_name_prefix

When USE_TABLE_NAME is specified in the Data Replication Console or in the apply.kafka.kafka_message_key parameter to use the mapped source table name as the Kafka topic name or message key name, you can use this parameter to also include a prefix in the topic and key names. For example, if you specify a prefix in this parameter and set the apply.kafka.kafka_table_name_include_schema parameter to 1, a Kafka topic name will have the following format: *prefix_schema_name*.*source_table_name*.

Default value: Null

apply.qadapter.cache_overflow_directory

Specifies the directory that Data Replication uses to write Least Recently Used (LRU) cache overflow files from the Queue Adapter.

Default value: *DataReplication_installation*/overflow.

apply.qadapter.cache_memory_size

For the Queue Adapter, specifies the maximum number of items in a single Least Recently Used (LRU) cache instance to keep in memory. When the number of cached items exceeds this limit, the overflow items are stored in the overflow directory that is set in the apply.qadapter.cache_overflow_directory parameter.

Valid values are 10 through 1,000,000,000

Default value: 1000

apply.qadapter.checkpoint_request_timeout

Specifies the time interval (in milliseconds) that the Applier waits for the Queue Adapter to validate that all data was successfully sent to a Kafka target before timing out. The Applier typically requests this checkpoint information at the end of each Apply cycle before the intermediate files are deleted. If the Queue Adapter does not validate that all data was sent to a Kafka target before this time elapses, the Queue Adapter reports a timeout error. The Applier then checks for error conditions, retries the request, or fails. Use this parameter and the apply.qadapter.checkpoint_retries parameter to provide enough time for the Applier to write all data to a Kafka target so that Data Replication can release the intermediate files for the Apply cycle.

Note: If Data Replication persistently issues an IDR-070218 error message and reports a syncKafkaEnvironment failure, try adjusting the apply.qadapter.checkpoint_request_timeout and apply.qadapter.checkpoint_retries parameter values.

Default value: 60000 (60 seconds)

apply.qadapter.checkpoint_retries

After a timeout error, specifies the maximum number of consecutive times that the Applier retries a request for the Queue Adapter to validate that all data was successfully sent to a Kafka target. If the Queue Adapter does not validate that all data was sent to a Kafka target after the maximum number of retries, the Applier ends with an error. For example, if the value of the apply.qadapter.checkpoint_request_timeout parameter is 60000 (60 seconds) and the value of this parameter is 6, then 6 minutes must expire before the Applier stops waiting for all data to be written to a Kafka target and ends with an error.

Note: If Data Replication persistently issues an IDR-070218 error message and reports a syncKafkaEnvironment failure, try adjusting the apply.qadapter.checkpoint_request_timeout and apply.qadapter.checkpoint_retries parameter values.

Default value: 5

apply.qadapter.create_missing_directories

When the directories that are specified in the apply.kafka.kafka_checkpoint_file_directory and apply.qadapter.cache_overflow_directory parameters do not already exist, determines whether to create these directories or report an access error. Valid values are:

- **0**. Report an access error, and do not create the directories.
- **1**. Create the directories.

Default value: 1

apply.qadapter.properties_file

Specifies a file that contains additional Queue Adapter properties for use by Informatica Global Customer Support or other Informatica personnel. The properties in this file are used to change or augment default Queue Adapter behavior with Data Replication. Edit this parameter only at the direction of Informatica Global Customer Support.

DB2 for Linux, UNIX, and Windows Parameters

apply.db2.create_lob_nologging

For configurations that have DB2 targets and any source other than DB2, determines the length of DB2 LOB columns that Data Replication creates when replicating CREATE TABLE and ADD COLUMN operations or when generating target tables based on source tables. The length determines whether logging can be enabled for these columns. Valid values are:

- **0**. Create LOB columns of 1 GB in length with the default LOGGED option. The LOGGED option enables logging.
- **1**. Create LOB columns of 2 GB in length with the NOT LOGGED option. The NOT LOGGED option disables logging.

  Note: DB2 for Linux, UNIX, and Windows does not support logging for LOB columns that are larger than 1 GB.

Default value: 0

db2.connection_additional_params

Specifies additional parameters for the ODBC connection string.

No default value

db2.direct_buffer_size

Specifies the buffer size, in bytes, that InitialSync uses for loading and unloading data when using the DB2 APIs.

Default value: 1048576 bytes

db2.initial_lsn

Specifies the LSN from which the Extractor starts extracting changes when you run the Extractor for the first time.

Default value: Current LSN

db2.max_failed_tasks

When you run InitialSync with DB2 sources and targets, Data Replication creates a separate process for each table. This parameter specifies the maximum number of processes that can fail before the InitialSync task fails.

Default value: 5

db2.node_name

Specifies the DB2 node name that the InitialSync component or Extractor uses to connect to a DB2 source. This parameter is required for DB2 sources.

No default value

db2.odbc.enable_decimal_period

If you use the ODBC driver to connect DB2 targets when running an Applier or InitialSync task, indicates whether the ODBC driver uses a period (.) as the decimal separator in float values that are loaded to the target. Options are:

- **0**. Preserve the decimal separator that is in the source float values when loading the float values to the target.
- **1**. Use a period (.) as a decimal separator in the target float values.

Default value: 1

db2.task_timeout

When you run InitialSync with DB2 sources and targets, Data Replication creates a separate process for each table. If a single process cannot open a pipe before the time limit that is specified in this parameter expires, the process fails. If you accept the default value of 0, Data Replication uses one of the following values:

- 5 seconds on Windows
- 20 seconds on Linux and UNIX

Default value: 0

extract.db2.checkpoint_size

Specifies the amount of transaction log, in MB, at which an extraction checkpoint is taken. If the extraction process fails, the next time you run it, it resumes from the last checkpoint.

Default value: 100 MB

extract.db2.db_alias_name

Specifies a database alias that the Extractor uses instead of the connection details on the **Source Database** tab to connect to a remote DB2 source database. This alias must be defined on the system where the Extractor runs.

If the parameter value is an empty string, the Extractor uses the connection details on the **Source Database** tab to connect to the DB2 source database.

Default value: empty string

extract.db2.enable_row_compression_support

Determines whether the DB2 Extractor requests the filtered log buffer for compressed rows from the db2ReadLog API. The Extractor needs the filtered log buffer to extract change data from tables that use row compression. Options are:

- **0**. Do not request the filtered log buffer for compressed rows. Use this option if none of the mapped source tables use row compression.
- **1**. Request the filtered log buffer for compressed rows. This option might degrade replication performance. Use this option only if one or more mapped source tables use row compression.

Default value: 0

Notes:

- When you create a replication configuration that maps a source table that uses row compression, the Data Replication Console sets this parameter to 1.
- To prevent data loss, set this parameter to 1 before enabling row compression for a mapped source table.

extract.db2.filter_option_forced

Indicates whether the DB2 Extractor always requests the DB2 filtered log buffer from the db2ReadLog API. Options are:

- **0**. Request the unfiltered log buffer. However, if the extract.db2.enable_row_compression_support runtime parameter is set to 1, the Extractor requests the filtered log buffer to get compressed records.
- **1**. Always request the filtered log buffer. The Extractor processes propagatable records from the filtered log buffer. These records do not contain DDL operations and LOB data that is not stored inline. This setting improves the Extractor performance.

  Note: If you enable DDL replication or map source LOB columns, the Data Replication Console sets the parameter to the default value of 0.

Default value: 0

extract.db2.lob_lookup_threads

Specifies the number of threads that the Extractor uses to extract LOB data from the source database. If you use the default value of 0, the Extractor ignores this parameter and extracts LOB data from the transaction logs.

Default value: 0

extract.db2.lob_truncate_size

Specifies the maximum amount of LOB data, in KB, that the Extractor extracts. If LOB data in a column is larger than the specified value, the data is truncated.

Default value: 64 KB

extract.db2.log_buffer_size

Specifies the buffer size, in kilobytes, that is used for parsing the transaction log. To improve performance, set this parameter to the estimated average log record size. A value that is too low or too high might significantly increase CPU consumption on the DB2 server. Lower values cause the Extractor to read the transaction log more frequently and might degrade Extractor performance.

Default value: 1024 KB

extract.db2.max_read_retries

Specifies the maximum number of times that the Extractor can retry a request to the DB2 API for the DB2 archive logs after the first request returns the log buffer in an invalid format. If all of the retries fail, the Extractor ends with an error.

Default value: 5

global.lsn_table_connection_attempts

Indicates the number of times the Extractor or InitialSync retries a query to the DBSYNC_SYNC_LSN service table after the initial query timeout interval elapses. The query timeout interval is specified in the global.lsn_table_connection_timeout parameter.

Default value: 3

global.lsn_table_connection_timeout

Indicates the number of seconds after which an Extractor or InitialSync query to the DBSYNC_SYNC_LSN service table will time out. After this interval elapses, the Extractor or InitialSync retries the query up to the number of attempts specified in the global.lsn_table_connection_attempts parameter.

Default value: 15

initial.db2.sync_lsn_table

Specifies the name of the service table. The Extractor uses the service table to determine the point in the transaction log from which the Applier should begin applying changes. The Extractor passes this information to the Applier.

You can change the default service table name when you save the configuration the first time. The Data Replication Console asks if you want to select a schema or to edit the service table name before it creates the table in the DB2 database.

After the configuration is saved, this parameter shows the full name of the service table in the following format: *schema_name.table_name*.

Default value: DBSYNC_SYNC_LSN

initial.db2.table_lock_enabled

Indicates whether to lock the source tables that the InitialSync component is processing. Options are:

- **1**. Lock.
- **0**. Do not lock.

InitialSync acquires a share lock on the source tables that are being synchronized to guarantee a consistent snapshot. However, you might want to prevent locking in some situations, for example, for test purposes.

Default value: 1

initial.db2.til

Specifies which transaction isolation level to use during the initial synchronization process. Options are:

- **0**. Default database transaction isolation level.
- **1**. Uncommitted read.
- **2**. Cursor stability.
- **3**. Read stability.
- **4**. Repeatable read.

Default value: 0

Greenplum Parameters

apply.enable_greenplum_table_redistribution

Indicates whether the Applier runs an ALTER TABLE *table* SET DISTRIBUTED *column*,... statement before replicating a DDL operation that adds a primary key constraint or unique index to a Greenplum target. Options are:

- **0**. Do not run the ALTER TABLE ... SET DISTRIBUTED statement.

  Important: If the target table uses the random distribution policy, the Applier cannot replicate DDL operations that add a primary key constraint or unique index.
- **1**. Run the ALTER TABLE ... SET DISTRIBUTED statement on a Greenplum target before applying a DDL operation that adds a primary key constraint or unique index. This option sets the distribution key columns to the unique constraint columns.

  Important: If the target table contains a large number of rows, this option can be resource intensive.

Default value: 1

Change the default value only at the request of Informatica Global Customer Support.

Microsoft SQL Server Parameters

extract.mssql.checkpoint_size

Specifies the amount of a transaction log, in megabytes, at which an extraction checkpoint is taken. If the extraction process fails, the next time you run it, it resumes from the last checkpoint.

Default value: 100 MB

extract.mssql.ignore_inactive_backup_locations

If you add shared backup log locations on the **Extract Range** tab, indicates whether the Extractor ends with an error when a location becomes temporarily unavailable. Options are:

- **0**. The Extractor ends with an error if it cannot access a backup log location.
- **1**. The Extractor continues processing even if it cannot access a backup log location.
  Use this option for Microsoft SQL Server sources that use Always On Availability Groups.

Default value: 0

extract.mssql.log_buffer_size

Specifies the buffer size, in megabytes, that the Extractor uses to parse the backup and online logs. Higher values might improve Extractor performance but require more memory to allocate log data. Lower values cause the Extractor to read the transaction log more frequently and might degrade Extractor performance.

Default value: 100 MB

extract.mssql.process_updates_as_updates

Indicates whether the Extractor handles each SQL Server Update as a pair of Delete and Insert operations or as a single Update. Options are:

- **0**. Handle each Update as a pair of Delete and Insert operations.
- **1**. Handle each Update as an Update. The Extractor writes a single Update operation to the intermediate files. As a result, the performance of Applier processing of Updates in SQL Apply mode is improved.

Default value: 1

Notes:

- Microsoft SQL Server 2008 R2 generates a pair of Delete and Insert operations for each Update. If you set this parameter to 1, the Extractor merges these Delete and Insert operations into a single Update operation. Other supported SQL Server versions process Updates as Updates. If you set this parameter to 0, the Extractor splits each Update operation into a pair of Delete and Insert operations.
- For configurations that include Merge Apply mappings, the Data Replication Console sets this parameter to 0.

extract.mssql.together_with_native_replication

Indicates whether the Extractor manages the secondary truncation checkpoint. Valid values are:

- **0**. Enable Extractor management of the secondary truncation checkpoint.
- **1**. Disable Extractor management of the secondary truncation checkpoint. Set this parameter to **1** in the following cases:
  - You run the Extractor that reads transaction logs and SQL Server Change Data Capture (CDC) or native transactional replication against the same database. In this case, SQL Server must manage the secondary truncation checkpoint and produce backup logs.

- You run two Extractors that read transaction logs from the same database. In this case, only one Extractor must manage the secondary truncation checkpoint.
- You run the Extractor that reads transaction logs and run the Microsoft SQL Server Backup task with the **Run the sp_repldone procedure** check box selected.

Important: After you disable the secondary truncation checkpoint management by the Extractor, ensure that the Extractor reads both the transaction and backup logs to prevent incomplete data capture.

This parameter is a synonym for the **Disable secondary truncation checkpoint** parameter that you can set on the **Runtime Settings tab** > **General view**.

Default value: 0

extract.use_db_connection

Indicates whether the Microsoft SQL Server Extractor is connected to the SQL Server source database. The Extractor must connect to the source database when reading online logs or working in continuous mode. Options are:

- **0**. Do not connect to the SQL Server source database.
- **1**. Connect to the SQL Server source database.

Default value: 1

mssql.enable_quoted_identifier

For Microsoft SQL Server sources and targets, indicates whether InitialSync adds the ODBC EnableQuotedIdentifiers=1 parameter to the connection string. When EnableQuotedIdentifiers is set to 1, SQL Server allows identifiers that are delimited by double quotation marks in Transact-SQL statements. Quoted identifiers can include reserved keywords and special characters that are not otherwise allowed in Transact-SQL statements. Options are:

- **0**. Do not add the EnableQuotedIdentifiers parameter to the connection string.
- **1**. Set the EnableQuotedIdentifiers parameter to 1 in the connection string to allow quoted identifiers.

Default value: 0

MySQL Parameters

mysql.connection_encoding

Specifies the connection-related character set that the InitialSync and Applier tasks use to load data to MySQL targets. The MySQL server uses this setting to convert the statements that are executed by the InitialSync and Applier tasks to the character set that is used by the database server.

The default connection-related character set for the InitialSync and Applier tasks is determined by the MySQL server character settings. Use this runtime parameter if the data that InitialSync unloads or that the Extractor extracts has a character set other than the default connection character set for the MySQL target. For a list of supported connection character sets for MySQL databases, see the MySQL documentation.

Netezza Parameters

apply.asyncloader.exit_on_odbc_error

Indicates whether the Applier is forced to end with an error after an ODBC error occurs on the Netezza target. Options are:

- **0**. The Applier completes normal processing of the ODBC error.
- **1**. The Applier is forced to end with an error.

Default value: 0

apply.asyncloader.reopen_pipe_timeout

If a Netezza target fails to open the pipe after an ODBC error, the Applier might continue to write change data to the pipe and then hang. In this case, Data Replication checks the pipe two times to determine if the Applier is still writing data to this pipe. If the apply.asyncloader.exit_on_odbc_error runtime parameter is set to 0, this parameter specifies the number of seconds between the first and second checks.

If the Applier is writing data to the pipe after the second check, Data Replication tries to open and close the pipe to complete normal processing of the error.

Default value: 1

Increase this parameter value if high loads occur on the Netezza target.

apply.netezza.connection_timeout

Specifies the number of seconds that the Applier waits for a response from the Netezza target. If this timeout period elapses without the response, the Applier ends with an error message.

Default value: 600 seconds

Increase this parameter value if high loads occur on the Netezza target.

nzload.ignorezero

Indicates whether the Applier uses the Netezza IGNOREZERO external table option to strip null characters (0x00) from extracted source data when applying the data to a character column on the Netezza target. The null characters can be null values within the source data or null terminator characters. Options are:

- **0**. Do not strip null characters from the data. The Netezza target returns an error if the Applier tries to apply data that includes null characters to a character column.
- **1**. Strip null characters from the data before applying the data to a character column. If a column contains null characters with other characters, the Applier forces Netezza to strip the null characters and then applies the other characters to the Netezza target column. For fixed-length target columns, the Applier pads the character string with blanks up to the fixed column length. If a source column contains only null characters, the Applier applies a null value to the Netezza target column.

  Caution: If the Applier applies a null value to a target column that is defined with the NOT NULL constraint, a constraint violation occurs.

Note: The InitialSync task truncates source data at the first null character that it encounters, regardless of this parameter setting.

Default value: 0

Oracle Parameters

apply.direct_load_for_audit_tables

For data replications in Audit Apply mode, indicates whether to use the native database load utility or the Oracle ODBC driver, which is delivered with the Oracle Client, to load data to audit log tables on the Oracle target. Options are:

- **0**. Use the ODBC driver for Oracle to load data to the audit log tables on the Oracle target.
- **1**. Use the native database load utility to load data to the audit log tables on the Oracle target.

Default value: 1

Note: For configurations that include Audit Apply mappings of tables with LOB columns, the Data Replication Console sets this parameter to 0.

apply.disable_target_triggers

For Oracle targets of 10.2.0.5 or a later 10.2 patch and for Oracle targets of 11.2.0.2 or a later 11.2 patch, indicates whether to disable suppressing triggers. Options are:

- **0**. Suppress triggers.
- **1**. Do not suppress triggers.

Default value: 0

apply.local_time_zone_hours

Specifies the hour value in the local time zone GMT offset for the source database. The Applier task uses this information to process TIMESTAMP WITH LOCAL TIMEZONE columns.

Default value: 0

apply.local_time_zone_minutes

Specifies the minute value in the local time zone GMT offset for the source database. The Applier task uses this information to process TIMESTAMP WITH LOCAL TIMEZONE columns.

Default value: 0

apply.oracle.use_returning_into

For Oracle targets, determines whether the Applier uses the RETURNING INTO clause to insert change data into LOB and LONG columns. Options are:

- **0**. Do not use the RETURNING INTO clause. The Applier cannot insert change data into LOB and LONG columns if the total row data size exceeds 4000 bytes. This setting can result in faster apply processing.
- **1**. Use the RETURNING INTO clause. The Applier can insert change data into LOB and LONG columns even if the total row data size exceeds 4000 bytes. This setting can result in slower apply processing.

Default value: 0

extract.max_flob_size

Determines the maximum buffer size that the Extractor allocates for Oracle LOB data when you perform row filtering based on a LOB value. If you enter a large value for this parameter, Data Replication allocates large object data in memory, which can degrade replication performance. If the LOB data is larger than the specified buffer size, the Extractor filters the data so that some of it fits in the buffer and skips the rest.

Default value: 12 KB

extract.oracle.check_size_of_parsed_logs

Indicates whether the Oracle Extractor ends with an error after processing a redo log file if the file size that is specified in the log header does not match the number of bytes that the Extractor processed. Options are:

- **0**. The Extractor logs a warning message and continues processing.
- **1**. The Extractor ends with an error.

Default value: 1

extract.oracle.print_logs_for_processing

Indicates whether the Extractor logs a message that lists the Oracle redo log files for processing to the Extractor execution log. In continuous mode, the Extractor logs this list at the beginning of each Extractor microcycle. If you run the replication schedule on demand or periodically, the Extractor logs this list when the Extractor task starts. Options are:

- **0**. Do not log the message that lists the redo log files for processing.

- **1**. Log the message that lists the redo log files for processing.

Default value: 0

extract.oracle.print_truncate_operations

Indicates whether the Extractor logs a message each time it encounters a TRUNCATE TABLE operation in the Oracle redo logs. The message includes the TRUNCATE TABLE statement. Options are:

- **0**. Do not log the message that reports the TRUNCATE TABLE statement.
- **1**. Log the message that reports the TRUNCATE TABLE statement.

Default value: 0

extract.oracle.rds_read_enable

Controls whether the Extractor can extract change data from online and archived redo logs for Amazon Relational Database Service (RDS) for Oracle instances in the cloud. Options are:

- **0**. The Extractor cannot process redo logs from Amazon RDS for Oracle instances.
- **1**. The Extractor can process redo logs from Amazon RDS for Oracle instances.

Default value: 0

extract.oracle.read_log_list_attempts

Specifies the maximum number of times the Extractor tries to get the list of Oracle redo log files for processing. If the redo logs do not pass integrity checking after the Extractor reaches the specified maximum number of attempts, the Extractor stops processing the redo logs.

Default value: 5

extract.oracle.skip_encrypted_tablespace_records

For Oracle sources that use Transparent Data Encryption (TDE) for tablespaces, determines the Extractor behavior when it cannot decrypt a record from an encrypted tablespace. Options are:

- **0**. Do not skip records from an encrypted tablespace that the Extractor cannot decrypt. In this case, the Extractor ends with an error.
- **1**. Skip records from an encrypted tablespace only if you did not open an Oracle PKCS #12-format wallet in the Data Replication Console.
- **2**. Skip records from an encrypted tablespace if you did not open an Oracle PKCS #12-format wallet or if the wallet that you opened in the Data Replication Console does not include a master key to decrypt these records.

Default value: 1

extract.oracle.skip_lob_data_for_table

Specifies the name of an Oracle source table for which you want the Extractor to replace LOB data with null values in the intermediate files. Use this parameter for a table that includes SecureFile LOB columns that use unsupported options, such as NOLOGGING or FILESYSTEM_LIKE_LOGGING. Otherwise, the Extractor ends with an error.

Default value: Empty

extract.oracle.threads_offline_at_start

Specifies a comma-separated list of the Oracle redo threads that are offline at the SCN that corresponds to the Start Point value for the Extractor. The Extractor starts processing change data records from an Oracle redo thread that is in this list only after a Thread Enable Marker (TEM) appears in the redo log. The Data

Replication Console populates this list of offline redo threads. Informatica recommends that you do not change the populated values for this parameter.

extract.oracle.threads_online_at_start

Specifies a comma-separated list of the Oracle redo threads that are online at the SCN that corresponds to the Start Point value for the Extractor. The Extractor starts processing change data records from Oracle redo threads that are in this list. The Data Replication Console populates this list of online redo threads. Informatica recommends that you do not change the populated values for this parameter.

extract.process_missing_columns

Determines the Oracle Extractor behavior after it encounters a source column that is missing from the configuration. Options are:

- **0**. The Extractor issues a parser exception. Subsequent Extractor behavior depends on the extract.stop_on_parsing_error runtime parameter setting.
- **1**. The Extractor writes change data for the missing column to an intermediate file.
  Use this option to correctly replicate ADD COLUMN ... DEFAULT operations that occur on the source. The Applier updates all of the existing rows in the mapped target table with the default value for the added column. If the target table contains many rows, this operation might take a long time.
- **2**. The Extractor does not write change data for the missing column to an intermediate file.
  Use this option to reduce overhead on Merge Apply targets after an ADD COLUMN ... DEFAULT operation on the source. The Applier does not update the existing rows in the mapped target table with the default value of the added source column.

Default value: 0

extract.stop_on_table_rename

Indicates whether the Extractor ends with an error after a mapped table is renamed in the Oracle source database. Options are:

- **0**. Log a warning message and continue Extractor processing. In this case, the Extractor continues extracting change data from the renamed table. The Applier continues to apply change data to the mapped target table. However, the target table is not renamed.
- **1**. End with an error.

Default value: 0

extract.use_strict_intermediate_file_size

For Oracle sources, determines whether to treat the value that is specified in the **Maximum size of each intermediate file** field on the **Runtime Settings** tab > **General** view as a soft limit or hard limit on the intermediate file size. Options are:

- **0**. The Extractor treats the value in the **Maximum size of each intermediate file** field as a soft limit on the intermediate file size. The Extractor checks the size of the intermediate file at each checkpoint. If the file size exceeds the specified limit, the Extractor writes the data and closes the intermediate file.
- **1**. The Extractor treats the value in the **Maximum size of each intermediate file** field as a hard limit on the intermediate file size. The Extractor estimates the size of the intermediate file when processing each change record. If the estimated file size exceeds the specified limit, the Extractor forces a checkpoint, writes the data, and closes the intermediate file.

Default value: 0

extract.verbose_ddl

Indicates whether to include debug messages that are related to Extractor processing of source DDL in the Extractor output. The messages provide information related to DDL changes that the Extractor parses from Oracle system tables. Use these messages to identify potential DDL processing errors. Options are:

- **0**. Do not include debug messages that are related to source DDL processing in the Extractor output.
- **1**. Include debug messages that are related to source DDL processing in the Extractor output.

Default value: 0

general.number_overflow_abort

For Amazon Redshift, DB2 for Linux, UNIX, and Windows, Greenplum, Microsoft SQL Server, Netezza, Teradata, and Vertica targets, if you use the general.number_overflow_verify parameter setting of 1, indicates whether the Applier and InitialSync end with an error if the whole number portion of an Oracle source NUMBER value exceeds the size of the target column. This situation is called a *numeric data overflow*. Options are:

- **0**. The Applier and InitialSync do not end with an error if a numeric data overflow occurs. Replace the number with the value that is specified in the general.number_overflow_replace parameter.
- **1**. The Applier and InitialSync end with an error if a numeric data overflow occurs.

Default value: 1

general.number_overflow_replace

For Amazon Redshift, DB2 for Linux, UNIX, and Windows, Greenplum, Microsoft SQL Server, Netezza, Teradata, and Vertica targets, if you use the general.number_overflow_verify setting of 1 and the general.number_overflow_abort setting of 0, specifies the replacement value for the Oracle NUMBER values that cause a numeric data overflow on the target. Valid values are any floating-point numbers that do not cause a numeric data overflow on the target.

When a numeric data overflow occurs, the Applier and InitialSync do not end with an error. Instead, the task replaces the NUMBER values that caused the overflow on the targets with the specified replacement value.

Default value: Null value

general.number_overflow_verify

For Amazon Redshift, DB2 for Linux, UNIX, and Windows, Greenplum, Microsoft SQL Server, Netezza, Teradata, and Vertica targets, indicates whether the Applier and InitialSync handle numeric data overflows for Oracle NUMBER columns that are replicated to target columns. Options are:

- **0**. Disable Applier and InitialSync handling of numeric data overflows. Allow the target database to handle numeric data overflow errors.
- **1**. Enable Applier and InitialSync handling of numeric data overflows. Use the general.number_overflow_abort and general.number_overflow_replace parameters to change the default handling of numeric data overflows on the target.

Default value: 1

initial.dblink.disable_global_names

If InitialSync uses database links (dblinks) for initial synchronization of Oracle sources and Oracle targets, indicates whether to override the Oracle GLOBAL_NAMES parameter value of true with false for the InitialSync targets. An override is required when the GLOBAL_NAMES parameter is set to true because InitialSync creates database link names that differ from the target database names. Options are:

- **0**. Do not change the GLOBAL_NAMES parameter setting for InitialSync sessions that have Oracle targets. Use this option only if the GLOBAL_NAMES parameter is set to false for the Oracle targets.

- **1**. Override the GLOBAL_NAMES parameter setting of true with false for InitialSync sessions that have Oracle targets. Use this option if the GLOBAL_NAMES parameter is set to true for the Oracle targets.

Default value: 0

Important: If the GLOBAL_NAMES parameter setting for an Oracle target is true, the Data Replication Console sets this initial.dblink.disable_global_names parameter to 1. If you set this parameter back to 0, InitialSync cannot create database links.

initial.oracle_clob_text_buffer_size

For Oracle sources, indicates whether InitialSync unloads CLOB data as character data of limited size. Enter a value greater than 0 to specify the buffer size, in bytes, that InitialScync can use to unload CLOB data as character data. Enter 0 to have InitialSync unload CLOB data as LOB data.

For Teradata and Netezza targets, this parameter is honored if the initial.enforce_odbc_load runtime parameter is set to 1. For Oracle targets, this parameter is honored if InitialSync does not use dblinks.

Default value: For configurations with Teradata targets, the default is the value of the global.lob_truncation_size runtime parameter. For configurations with any other target type, the default is 0.

initial.oracle.parallel_sample_percentage

For Oracle sources, specifies the percentage of the total number of rows in a source table that InitialSync randomly samples to get row IDs for distributing the table rows across multiple InitialSync subtask threads. InitialSync executes a SELECT statement with a WHERE clause that uses the sampled row IDs to define a range of rows to be processed on a thread. In this manner, Data Replication uses the sampled row IDs to distribute all of the table rows across InitialSync threads for multithreaded load processing. For example, if you enter 2 percent for a table that contains 100 rows and the sampled row IDs are 30 and 60, Data Replication could distribute the table rows as follows:

- Thread1 processes rows 0-30.
- Thread2 processes rows 31-60.
- Thread3 processes rows 61-100.

Note: The number of InitialSync subtask threads is limited by the initial.oracle.parallel_subtask_limit runtime parameter.

For more information about using multiple InitialSync threads, see “Considerations for Running InitialSync” on page 274.

Default value: 0

If you use the default value of 0, InitialSync processes all table rows with a single thread. Use the default value of 0 for source tables for which InitialSync does not support multithreaded processing, including Oracle source tables that have subpartitions and tables that have virtual columns associated with Tcl scripts or SQL expressions.

Tip: Use the initial.oracle.parallel_sample_seed runtime parameter to select a reproducible sample of Oracle source table rows and row IDs for distributing table rows across multiple InitialSync threads, each time you run InitialSync. This way, you can better compare InitialSync performance across multiple executions that use different initial.oracle.parallel_sample_percentage parameter settings or different Data Replication versions.

initial.oracle.parallel_sample_seed

If you set the initial.oracle.parallel_sample_percentage parameter to a positive value, specifies the SEED value in the SELECT…SAMPLE statement that InitialSync uses to select a reproducible sample of Oracle source table rows and row IDS for distributing table rows across multiple InitialSync threads, each time you run InitialSync. By using a reproducible sample of rows, you can better compare InitialSync performance across multiple

executions that use different initial.oracle.parallel_sample_percentage parameter settings or different Data Replication versions.

Valid values are 0 through 2147483647

Default value: 0, which provides a different distribution of rows across InitialSync threads each time InitialSync runs.

initial.oracle.parallel_subtask_limit

If the initial.oracle.parallel_sample_percentage runtime parameter is set to a non-zero value, specifies the maximum number of subtask threads that InitialSync can use to load data in parallel to the targets that you are materializing from Oracle sources.

Default value: 4

initialsync.oracle.check_corruption_event

Indicates whether to detect a corrupted Oracle source when running an InitialSync task. This option uses event 10200 to indicate the error.

Default value: false

initialsync.oracle.no_parallel_select

For Oracle sources, indicates whether to disable parallel query processing when running an InitialSync task. Options are:

- **false**. Use parallel query processing for Oracle sources.
- **true**. Disable parallel query processing for Oracle sources for InitialSync.

Default value: false

initialsync.oracle.skip_corruption_event

Indicates whether to skip corrupted blocks in an Oracle source database when running the InitialSync task. This option uses events 10231 and 10233 to indicate the error.

Default value: false

initialsync.oracle.use_dblink

For Oracle-to-Oracle initial synchronization only, indicates whether Data Replication can use database links (dblinks). The use of dblinks can improve performance. Options are:

- **1**. Enables Data Replication to use dblinks.
- **0**. Prevents Data Replication from using dblinks.

Default value: 1

oracle.replace_empty_string_characters

If you replicate source columns that contain empty strings to Oracle targets, specifies the character that the Applier and InitialSync use to replace the empty strings when writing data to the corresponding target columns.

You can enter any string that is valid for the target column datatype. The default value is a space, which causes the Applier and InitialSync to replace the empty strings with the space character on the target.

For this parameter to be honored, you must set the oracle.replace_empty_string_enabled parameter to 1.

Default value: Space character

oracle.replace_empty_string_enabled

For Oracle targets, indicates whether the Applier and InitialSync replace empty strings from the source with a character value when writing the strings to the target. Options are:

- **0**. Do not replace empty strings from the source with a character value on the target. The Applier and InitialSync replicate the empty strings as null values to the target.
- **1**. Replace empty strings from the source with a character value on the target. The oracle.replace_empty_string_characters runtime parameter specifies the replacement character that the Applier and InitialSync use. Use this option if you map source columns that include empty strings to NOT NULL target columns to avoid NOT NULL constraint violations.

Default value: 0

oracle.replace_empty_string_not_null_only

For Oracle targets, if you set the oracle.replace_empty_string_enabled parameter to 1, indicates whether the Applier and InitialSync replace empty strings from the source with a replacement character when writing data to all of the target columns or only to NOT NULL target columns. Options are:

- **0**. Replace the empty strings with a replacement character in both NOT NULL and NULL target columns.
- **1**. Replace the empty strings with a replacement character only in NOT NULL target columns.

You can specify the replacement character in the oracle.replace_empty_string_characters parameter.

Default value: 1

oracle.truncate_char_to_4000

For Oracle targets, indicates whether the Applier truncates character data to 4,000 bytes if the corresponding source tables have character data that is larger than 4,000 bytes. Options are:

- **0**. Do not truncate character data.
- **1**. Truncate character data to 4,000 bytes.

Default value: 0

Teradata Parameters

apply.teradata.td_max_sessions

Specifies the maximum number of sessions for TPT libraries that an Applier thread can use for loading data to a target. The value for this parameter must be less than or equal to the number of AMPs (Access Module Processors) on the Teradata target database.

Default value: 8

apply.teradata.td_tenacity_hours

Adjusts the TENACITY runtime option for the TPT Load or Update operator when you run the Applier task. This parameter specifies the number of hours that the TPT Load or Update operator tries to log on when the maximum number of load jobs is already running on the Teradata database.

For more information about the TENACITY option, see the Teradata documentation.

Default value: 0

initialsync.teradata.td_max_sessions

Specifies the maximum number of sessions for TPT libraries that an InitialSync thread can use for loading data to a target. The value for this parameter must be less than or equal to the number of AMPs on the Teradata target database.

Default value: 8

initialsync.teradata.td_tenacity_hours

Adjusts the TENACITY runtime option for the TPT Update operator when you run the InitialSync task. This parameter specifies the number of hours that the TPT Update operator tries to log on when the maximum number of load jobs is already running on the Teradata database.

For more information about the TENACITY option, see the Teradata documentation.

Default value: 0

teradata.disable_utf8_load

Indicates whether InitialSync and the Applier load source character data to a Teradata target by using UTF-8 encoding. Options are:

- **0**. Use UTF-8 encoding when loading source character data to the target.
- **1**. Do not use UTF-8 encoding when loading source character data to the target. This setting might improve replication performance.

  Note: Specify 1 if source character data includes only Latin characters.

Default value: 0

teradata.stream_load_hook

If the Applier uses the TPT Stream operator, indicates whether the Applier locks Teradata target tables before loading change data. Options are:

- **0**. Lock the target tables before loading change data.
- **1**. Do not lock the target tables before loading change data.

Default value: 1

teradata.stream_operator_enabled

Indicates the type of TPT loader that the Applier uses to load change data to Teradata targets. Options are:

- **0**. Always use the TPT Load operator to apply change data to Teradata targets in Merge Apply mode. Always use the TPT Update operator to apply change data to Teradata targets in Audit Apply mode.
- **1**. Use the TPT Stream operator or the TPT Load or Update operator to load change data to Teradata targets, depending on the amount of change data. If the amount of change data from a source table is greater than the value that is specified by the teradata.threshold_table_size parameter, the Applier uses the TPT Load operator in Merge Apply mode and uses the TPT Update operator in Audit Apply mode. If this amount is less than the teradata.threshold_table_size value, the Applier uses the TPT Stream operator in both Audit Apply and Merge Apply modes.

Default value: 1

teradata.stream_operator_max_connections

Specifies the maximum number of TPT Stream operator connections to a Teradata target that the Applier can use.

Default value: 4

teradata.td_trace_level

Specifies the tracing level for the Teradata driver. Options are:

- **TD_OFF**. Disable Teradata driver trace messages.
- **TD_OPER**. Enable trace messages for driver-specific activities.
- **TD_OPER_CLI**. Enable trace messages for driver activities that involve CLIv2.

- **TD_OPER_NOTIFY**. Enable trace messages for driver activities that involve the Notify feature.
- **TD_OPER_OPCOMMON**. Enable trace messages for driver activities that involve the operator common library.
- **TD_OPER_ALL**. Enable all driver-level trace messages.

Default value: TD_OFF

The Teradata driver writes trace messages to the log file that is specified by the teradata.td_trace_output_name runtime parameter.

teradata.td_trace_level_infr

Specifies the tracing level for the Teradata infrastructure. Options are:

- **TD_OFF**. Disable trace messages for Teradata infrastructure.
- **TD_OPER**. Enable trace messages for infrastructure-specific activities.
- **TD_OPER_CLI**. Enable trace messages for infrastructure activities that involve CLIv2.
- **TD_OPER_NOTIFY**. Enable trace messages for infrastructure activities that involve the Notify feature.
- **TD_OPER_OPCOMMON**. Enable trace messages for infrastructure activities that involve the operator common library.
- **TD_OPER_ALL**. Enable all infrastructure-level trace messages.

Default value: TD_OFF

The Teradata infrastructure writes trace messages to the log file that is specified by the teradata.td_trace_output_name runtime parameter.

teradata.td_trace_output_name

Specifies the name of the output file that includes the TPT trace messages. Data Replication uses the following naming pattern to generate this file in the *DataReplication_installation*/output directory:

```
file_name.schema_name.table_name.PID.timestamp
```

You can configure the tracing level of the Teradata driver by using the teradata.td_trace_level and teradata.td_trace_level_infr runtime parameters.

Default value: tpttrace. The full default name for the TPT trace log is tpttrace.*schema_name*.*table_name*.*PID*.*timestamp*.

teradata.threshold_table_size

Specifies the maximum amount of change data, in bytes, that the Applier can apply to a Teradata target table by using the TPT Stream operator in Merge Apply or Audit Apply mode. If the amount of change data exceeds this limit, the Applier applies change data to the target by using either the TPT Load operator in Merge Apply mode or the TPT Update operator in Audit Apply mode.

Tip: To determine the size of change data, select the configuration from the **Server Manager** > **Configs** view in the Data Replication Console and click the **Intermediate Files** icon button. The **Intermediate Files for the Configuration <configuration_name>** dialog box appears. For each file listed in the **Intermediate Files** box on the left, the amount of extracted data, in bytes, is displayed in the **Transactions** box on the right by transaction.

Data Replication compresses the size of intermediate files by default. You can edit this setting and adjust the maximum size, in megabytes, of intermediate files on the **Runtime Settings** tab > **General** view.

Default value : 1048576

teradata.tpt_thread_safe

Indicates whether to use additional mutual exclusion mechanisms for thread-unsafe tasks when an Applier or InitialSync task loads data to Teradata. Options are:

- **0**. Do not use additional mutual exclusion mechanisms for thread-unsafe tasks.
- **1**. Use additional mutual exclusion mechanisms for thread-unsafe tasks.

Default value: 0

Common Parameters

apply.buffer_size_for_split_records

Specifies the maximum size of the buffer, in bytes, that the Applier uses to store data for a long-running transaction or for a table that is queued for apply processing in Merge Apply mode. When this buffer size limit is exceeded, the Applier flushes data from the buffer to a temporary spill file and then writes any additional change data to the spill file. The Applier creates a buffer for each long-running transaction or for each table that is queued for apply processing in Merge Apply mode.

Default value: 1048576 bytes

apply.conflict_resolution_json_offset

If you use the apply.conflict_resolution_pretty_json parameter, specifies the indentation level for the JSON log files. Data Replication uses the specified number of spaces to indent JSON elements.

Default value: 1

apply.conflict_resolution_pretty_json

Indicates whether to use the indented format for the JSON log file to which the Applier writes information about detected conflicts. You specify a path to this log file on the **Runtime Settings** tab > **General** view. Options are:

- **0**. Use a single-line format for the JSON log files. Use this option to reduce the size of the log files.
- **1**. Use an indented format for the JSON log files. The indentation level is determined by the apply.conflict_resolution_json_offset parameter.

Default value: 1

apply.force_log_table_suffix_for_audit

For configurations that include multiple targets, indicates whether the Applier alters the SQL apply statements for secondary targets that use Audit Apply mode to append the audit log suffix to the table names on those targets. The audit log suffix is specified in the **Log table suffix for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view and usually used for Merge Apply. Options are:

- **0**. Do not append the audit log suffix to the table names on the secondary targets that use Audit Apply mode. With this option, the names of the audit log tables on the secondary targets match the names of the regular target tables on the primary target. This option is typically used for configurations that replicate data from one primary target to multiple secondary targets in different databases.

- **1**. Append the audit log suffix to the table names on the secondary targets that use Audit Apply mode. With this option, the audit log table names on the secondary targets are composed of the primary target table names followed by the audit log suffix. This option is used in special situations where you need to use the audit log suffix to distinguish the audit log tables on secondary targets that use Audit Apply mode.

  Important: You can use Audit Apply mode for a secondary target to add audit log tables in the primary target database that use the same schema as the regular target tables in the same database. In this case, if you use Merge Apply mode for the primary target, the audit log tables for Merge Apply mode must have a different schema. Specify a schema that includes audit log tables for Merge Apply mode in the **Log table schema for merge apply** field on the **Runtime Settings** tab > **Calculated Columns** view.

Default value: 0

apply.libpq_client_encoding

For Greenplum and Vertica targets, specifies the original source character encoding for data that Data Replication applies using the libpq library in Audit Apply and Merge Apply modes. InitialSync also uses this setting to specify original source character encoding for data that it loads using the libpq library. This parameter is equivalent to setting the PGCLIENTENCODING environment variable. For example, if the Oracle source system character set is WE8ISO8859P1, set this parameter to LATIN1. For more information about available character sets for a target, see your target database documentation.

No default value.

apply.libpq_custom_connection_string

For Greenplum, Vertica, and PostgreSQL targets, specifies a custom connection string for the libpq library. This parameter overrides the automatically generated libpq connection string, which has the following format:

```
host=target_host_name
port=target_port
user=target_user
password=target_password
dbname=target_database_name
```

No default value.

apply.load_sessions_per_load_pass

Specifies the number of target tables that the Applier can load concurrently. By default, Data Replication uses this parameter only for Netezza, Vertica, and Teradata targets. To use this parameter for other target types, set the apply.loader_multiple_load_passes parameter to 1.

If you use the value of -1 for this parameter, the Applier determines the number of tables that it can load to the target concurrently based on the target database type. The Applier uses the following criteria:

- For Netezza, the number of tables equals the number of Applier threads. Maximum is 20 tables.
- For Vertica, the number of tables equals the number of Applier threads multiplied by 4. Maximum is 20 tables.
- For other target types, the number of tables equals the number of Applier threads. No maximum limit applies.

Default value: -1

Important: To change this value for Netezza targets, Informatica recommends that you clear the **Create output as pipes instead of files** option on the **Runtime Settings** tab > **General** view. If this option is selected, each Applier thread can load only a single audit log table at a time.

apply.loader_multiple_load_passes

Indicates whether the number of target tables that the Applier loads concurrently is limited. Options are:

- **0**. The Applier loads data to all of the target tables concurrently. No limit on the number of tables is in effect.
- **1**. The Applier loads data to a limited number of target tables concurrently. The number of tables is limited by the apply.load_sessions_per_load_pass runtime parameter.

Default value: 0. However, for Netezza, Vertica, and Teradata targets, Data Replication always uses the parameter value of 1 because the number of concurrent loader tasks on these targets is limited.

apply.loader_tx_map_size_rollback_key

Specifies the number of slots in a hash structure that is used to process rollback-to-savepoint records during each apply cycle. Increase this value if rollback-to-savepoint operations are common and process a lot of records.

Default value: 262144

apply.loader_tx_map_size_tx

Specifies the number of slots in a hash structure that is used to hold committed transactions during each apply cycle. For better performance, set this value to one-tenth of the number of opened or closed transactions in each apply cycle.

Default value: 500000

apply.merge.double_precision

The Applier rounds 8-byte floating-point numbers, such as DOUBLE and 8-byte FLOAT values, from the source before loading them to Merge Apply targets. This parameter specifies the number of digits after the decimal point that the Applier preserves in the 8-byte floating-point numbers that are applied to the target.

Default value: 15

apply.merge.float_precision

The Applier rounds 4-byte floating-point numbers, such as 4-byte FLOAT and REAL values, from the source before loading them to Merge Apply targets. This parameter specifies the number of digits after the decimal point that the Applier preserves in the 4-byte floating-point numbers that are applied to the target.

Default value: 8

apply.performance_degradation_warning_percentage

Specifies the percentage degradation of the average SQL statement processing time that must occur before Data Replication logs a warning. You can specify the intervals at which Data Replication calculates the average SQL statement processing times in the apply.print_statistics_interval and initial.print_statistics_interval parameters.

Note: Data Replication does not log a performance degradation warning if the average SQL statement processing time is less than the value that is specified by the apply.performance_degradation_warning_threshold parameter.

Default value: 10%

apply.performance_degradation_warning_step

Specifies the number of performance readings that must indicate degradation of the average SQL statement processing time before Data Replication logs a warning. You can specify the intervals at which Data Replication calculates the average SQL statement processing times in the apply.print_statistics_interval and initial.print_statistics_interval parameters.

Note: Data Replication does not log a performance degradation warning if the average SQL statement processing time is less than the value that is specified by the apply.performance_degradation_warning_threshold parameter.

Default value: 3

apply.performance_degradation_warning_threshold

Specifies the minimum average SQL statement processing time, in milliseconds, over an apply cycle for which Data Replication logs a performance degradation warning.

Default value: 5 milliseconds

apply.post_apply_script_enabled

Indicates whether the Applier runs post-apply SQL statements or stored procedure calls after each apply cycle on a target. Post-apply processing is supported for configurations that have Greenplum, Netezza, Oracle, Vertica, or Teradata targets and that use Audit Apply or Merge Apply mode. You must specify the post-apply SQL statements or stored procedure calls in the `%DBSYNC_HOME%/uiconf/afterapply.xsl` file. Options are:

- **0**. Disable post-apply processing.
- **1**. Enable post-apply processing.

Default value: 1

apply.print_statistics_interval

For configurations with targets that use ODBC drivers, specifies the number of seconds that must elapse before Data Replication updates the number of processed rows in the Applier output log. If you specify 0, Data Replication does not include the number of processed rows in the Applier output log.

Default value: 300 seconds

apply.process_intermediate_size_per_job

Specifies the maximum total size of all intermediate files, in megabytes, that Data Replication processes during a single apply cycle. Data Replication always processes entire intermediate files. Data Replication never splits an intermediate file to avoid exceeding the maximum total size that is specified in this parameter. You control the maximum size of a single intermediate file by setting the **Maximum size of each intermediate file** option on the **Runtime Settings** tab > **General** view.

Each apply cycle ends with a single commit by default except in certain situations when you use Applier multi-threaded processing. For more information, see "Commit Processing and Target Constraints with Applier Multi-Threaded Processing" on page 40. If you specify smaller values for this parameter, Data Replication apply processing reduces the transaction size and increases the commit frequency on the target. If you specify 0, Data Replication processes all available intermediate files within a single apply cycle.

Note: Data Replication always processes at least one intermediate file during a single apply cycle, even if the intermediate file size exceeds the maximum total size that is specified in this parameter.

Default value: 256 MB

apply.realtime_clean_metadata_every_n_cycles

For continuous replication, specifies the number of apply microcycles that the Applier runs before it removes information about the processed intermediate files from the Applier SQLite database. This parameter is for use only by Informatica Global Customer Support. Do not modify it.

Default value: 1000 microcycles

apply.recovery_enabled

Indicates whether to enable recovery based on information in the recovery table for apply processing. Options are:

- **0**. Disable recovery based on information in the recovery table. When you save a new configuration in the Data Replication Console, this parameter is transparently set to 0 if you clicked **No** in the **Enter Recovery Table Name and Schema** dialog box to not create a recovery table with the specified name.
- **1**. Enable recovery based on information in the recovery table. When you save a new configuration in the Data Replication Console, this parameter is transparently set to 1 if you clicked **Yes** in the **Enter Recovery Table Name and Schema** dialog box to create a recovery table. Ensure that the apply.recovery_table parameter specifies the schema and table name of the recovery table.

For database target types other than Cloudera, Flat File, Hortonworks, and Kafka, Informatica recommends that you enter 1 to enable recovery for apply processing. For Cloudera, Flat File, Hortonworks, and Kafka targets that do not use a recovery table, enter 0.

Default value: 0 for the Cloudera, Flat File, Hortonworks, and Kafka targets, 1 for other targets.

apply.recovery_table

Specifies the schema and table name of the recovery table on the primary target. Only target types other than Cloudera, Flat File, Hortonworks, and Kafka use recovery tables. For the recovery table to be used, ensure that the apply.recovery_enabled parameter is set to 1.

In the Data Replication Console, when you save a new configuration, the Console prompts you to specify the schema and table name of the recovery table. You can accept the default values or override them. The default schema is the database user, and the default table name is IDR_RECOVERY. If you click **Yes**, this parameter is populated with the full recovery table name. The Applier uses this parameter value to create the recovery table. If you click **No**, the recovery table is not created and the apply.recovery_enabled parameter is set to 0. If you want to use a recovery table later, you can create it in one of the following ways:

- Manually create the table by using the DDL in Appendix F, “DDL Statements for Manually Creating Recovery Tables” on page 455. Then enter the schema and table name of the table that you created in this parameter and set the apply.recovery_enabled parameter to 1, before saving the configuration again.
- In the Data Replication Console, set the apply.recovery_enabled runtime parameter to 1 and save the configuration again. The Console prompts you for the schema and table name. After you specify these names and click **Yes**, the Console enters the full *schema.table_name* value in this parameter. When you start the Applier later, it will create the recovery table.

Note: To specify recovery tables for secondary targets in a configuration that has multiple targets, use the **Recovery table name** field on the **Routing** tab > **Override Apply** subtab.

Default value: IDR_RECOVERY

apply.script_engine_enabled

Indicates whether the Tcl Script Engine executes the following Tcl scripts that are located in the *DataReplication_installation*/support/Tcl directory during apply processing:

- Execute dbs_main_start.tcl when the Applier task starts.
- Execute dbs_main_end.tcl when the Applier task ends.
- Execute dbs_cycle_start.tcl when an apply cycle starts.
- Execute dbs_cycle_end.tcl when an apply cycle ends.

Options are:

- **0**. Do not execute these Tcl scripts during apply processing.

- **1**. Execute these Tcl scripts during apply processing.

Default value: 0

apply.skip_if_pk_not_defined

Indicates whether the Applier is forced to end with an error if it encounters an Update or Delete record that does not contain undo values for all of the primary key columns. Options are:

- **0**. The Applier ends with an error.
- **1**. In SQL Apply and Merge Apply modes, the Applier skips any Update or Delete record that does not include undo values for all of the primary key columns and continues apply processing. This setting can result in data inconsistencies.

  Note: In Audit Apply mode, the Applier applies these records to the target instead of skipping them.

Default value: 0

apply.skip_oracle_piecewise_operations

Indicates whether the Applier skips Oracle piecewise operations.

Data Replication supports Oracle piecewise operations only for Oracle targets in Audit Apply mode if apply.direct_load_for_audit_tables is set to 0 and in SQL Apply mode. This parameter determines the Applier behavior when processing change records that contain piecewise operations in the following cases:

- For target databases other than Oracle.
- For Oracle targets in Merge Apply mode.
- For Oracle targets in Audit Apply mode if apply.direct_load_for_audit_tables is set to 1.

Options are:

- **0**. Do not skip Oracle piecewise operations. If piecewise operations occur, the Applier ends with an error.
- **1**. Skip Oracle piecewise operations.

Default value: 0

apply.sql_statement_text_id_cache_optimization_enabled

For source tables mapped in SQL Apply mode, controls whether the Applier caches the SQL statements that it generates for applying data to target tables in memory. Enable caching of these SQL statements to improve Applier performance. Options are:

- **0**. Do not cache the SQL statements for applying data to the target tables.
- **1**. Cache the SQL statements for applying data to the target tables.

Default value: 1

apply.threshold_time_for_open_transaction

Specifies the number of minutes that a transaction must be open before the Applier logs a warning message about the open transaction. The Applier compares this threshold time to the time the transaction has been open, which is calculated as the difference between the time of the last record from the open transaction and the time of the last processed record from any transaction.

Default value: 60 minutes

apply.trace_db_execution

For Oracle targets, indicates whether to use Oracle SQL tracing. This tracing is enabled using SQL event 10046 with trace levels 1, 4, 8, or 12. Options are:

- **0**. Do not use SQL tracing.
- **1**. Use standard SQL trace level 1. Provides statistics for parse, execute, fetch, commit and rollback database calls.
- **4**. Use SQL trace level 4. This option displays level 1 information and additional information about bind variables.
- **8**. Use SQL trace level 8. This option displays level 1 information and additional information about wait time.
- **12**. Use SQL trace level 12. This option enables trace levels 4 and 8 simultaneously.

Default value: 0

apply.verbose_ddl

Indicates whether to include the DDL statements that the Applier uses to apply source DDL changes to the target in the Applier output. The Applier translates the source DDL statements to DDL statements that the target database can process. You can include these translated DDL statements in the Applier output to determine the specific DDL statements that the Applier used to apply structural changes on the target. Options are:

- **0**. Do not include the SQL statements for the target that correspond to source DDL changes in the Applier output.
- **1**. Include the SQL statements for the target in the Applier output.

Default value: 0

apply.verbose_level

Specifies the level of verbose output for the Applier process. Larger values provide more detailed output. The specific output information depends on the Applier type. Valid values are 0 to 4.

Default value: 0

apply.verbose_process_long_running_txns

Indicates whether the Applier logs messages that are related to processing memory buffers and temporary spill files for change data that belongs to long-running open transactions. Options are:

- **0**. Do not include the messages that are related to processing memory buffers and temporary spill files for long-running transactions in the Applier output.
- **1**. Include the messages that are related to processing memory buffers and temporary spill files for long-running transactions in the Applier output.

Default value: 1

apply.verbose_print_every_n_records

Determines the frequency at which Data Replication updates progress when the Applier task is running. When the specified number of records are reached, Data Replication updates progress information.

Default value: 1000000 records

apply.verbose_process_queued_tables

Indicates whether the Applier logs messages that are related to processing memory buffers and temporary spill files for queued source tables. The Applier cannot apply change data from these tables during the current apply cycle because of target loader limitations. Options are:

- **0**. Do not include the messages that are related to processing memory buffers and temporary spill files for queued tables in the Applier output.
- **1**. Include the messages that are related to processing memory buffers and temporary spill files for queued tables in the Applier output.

Default value: 0

apply.verify_modified_row_count

Indicates whether Data Replication verifies that the number of rows that the Applier expected to process for Inserts, Updates, and Deletes during the apply cycle matches the actual count of rows for these SQL changes that were applied. Options are:

- **0**. Do not verify that the expected row count matches the actual row count.
- **1**. Verify that the expected row count matches the actual row count.
  A row count mismatch might occur in the following situations:
  - Data Replication replicates a row for an Insert operation, but an Insert row was already written to the target.
  - Data Replication replicates a row for a Delete or Update operation, but no rows match the WHERE condition that is used when applying changes to the target.

  With this setting, the apply cycle is rolled back and a new apply cycle begins with row-by-row processing. Data Replication prints the SQL statements and bind variable values with the mismatched row count to the log. The Applier then reports the mismatch and either stops or continues depending on the apply.verify_modified_row_count_abort_on_mismatch parameter setting.

This type of verification works only for SQL Apply table mappings. It does not work for Audit Apply or Merge Apply mappings.

Default value: 1

apply.verify_modified_row_count_abort_on_mismatch

If you set the apply.verify_modified_row_count parameter to 1, use this parameter to control whether the Applier task stops or continues when a mismatch occurs between the expected count of Insert, Update, and Delete rows and the actual row count. Options are:

- **0**. Report an error and continue apply processing.
- **1**. Stop apply processing.

Default value: 1

data_direct_optimize_prepare

Specifies the value for the OptimizePrepare (OP) attribute in the connection string that is generated for the DataDirect ODBC drivers. The OptimizePrepare attribute determines whether the driver creates stored procedures on the database server for calls to the ODBC SQLPrepare function. Options are:

- **0**. Create stored procedures for each call to the SQLPrepare function. This setting might degrade performance when processing SQL statements that do not contain parameters.
- **1**. Create stored procedures only if the SQL statement includes parameters. If the SQL statement does not include parameters, the driver caches the SQL statement and runs it directly when calling the SQLExecute function.

- **2**. Do not create stored procedures. The driver caches the SQL statement and runs it directly in the SQLExecute function. The driver reports syntax errors during the call to the function or when the SQLExecute function runs.
- **3**. Do not create stored procedures. The driver caches the SQL statement and runs it directly in the SQLExecute function. However, the driver reports syntax errors during the call to the function or when the SQLPrepare function runs.

Default value: 0

extract.log_wait_cycle_time

Specifies the number of seconds that the Extractor waits during a cycle for archived log files from each Oracle RAC to appear before starting a new cycle.

Default value: 3 seconds

extract.log_wait_cycles

Specifies the number of cycles that the Extractor runs while waiting for Oracle RAC archived logs before it exits with an error message.

Default value: 40

extract.log_wait_print_warning_cycles

Specifies the number of cycles that the Extractor runs while waiting for Oracle RAC archived logs before it prints a warning message that reports its wait status.

Default value: 5

extract.process_old_logs

For Microsoft SQL Server and Oracle sources, indicates whether the Extractor reads all available transaction logs. Options are:

- **0**. Start reading the logs from the Start Point value and skip any older logs. The default Start Point value is calculated by the Data Replication Console. You can override the default Start Point value to customize Extractor processing.
- **1**. Ignore the Start Point value and process all available log files.

Default value: 0

extract.rawdevice_offset

On Linux and AIX operating systems only, if you use raw devices to store data for an Oracle source database, controls the offset from the beginning of the raw device storage at which the Oracle Extractor begins reading data. The offset is expressed in bytes. The default offset depends on the operating system type. Valid values are the integers 0 through 2147483647. Data Replication interprets any negative integer value for this parameter as the default value.

Default value: -1. On Linux, the default value causes the offset to be 0 bytes. On AIX, the default value causes the offset to be 4096 bytes.

extract.show_throughput

For Oracle sources, the Extractor shows the amount of the redo logs in bytes that is processed per time unit. Options are:

- **0**. Show the amount of redo logs processed per time unit.
- **1**. Do not show the amount of redo logs processed per time unit.

Default value: 0

extract.stop_on_parsing_error

Determines the Extractor behavior when the Extractor cannot parse a transaction record in the log. Options are:

- **0**. Skip an inconsistent log record and continue processing. The Extractor logs a message for each skipped record and ends successfully if no other errors occur.
- **1**. Log an error and stop processing the transaction log.

When a parsing error occurs, the Oracle Extractor also creates a binary file with the transaction record and a .LOG file in the *DataReplication_installation*/dump subdirectory.

Default value: 1

extract.throughput_window

The number of measurements that Data Replication uses to calculate Extractor throughput. Throughput is reported in an Extractor message in the output log. Valid values are positive integers, beginning with 1.

Default value: 5

extract.verbose_level

Specifies the level of verbose output for the Extractor process. Larger values provide more detailed output. The specific output information depends on the Extractor type. Valid values are 0 to 4.

Default value: 0

global.encrypt_algorithm

Specifies the encryption algorithm that Data Replication uses to encrypt the intermediate files. This setting is also configurable in the Data Replication Console on the **Runtime Settings** tab > **General** view. In the Encryption Algorithm box, options in the **Type** list correspond to the options for this parameter. Options are:

- **0**. Do not encrypt the intermediate files.
- **1**. Use the Triple DES algorithm.
- **2**. Use the AES algorithm with 128-bit key.
- **3**. Use the AES algorithm with 192-bit key.
- **4**. Use the AES algorithm with 256-bit key.

Default value: 0

global.fallback_connection

When an attempt to connect to an Oracle source or target fails, indicates whether the Extractor and Applier rebuild the ODBC connection string and try to connect to the source or target again. Options are:

- **0**. If an attempt to connect to an Oracle source or target fails, the replication tasks do not try to connect to the source or target again. The tasks end with an error.

- **1**. If an attempt to connect to an Oracle source or target fails, the replication tasks rebuild the ODBC connection string and try to connect to the source or target again. The replication tasks rebuild the connection string as follows:
  - If the replication configuration does not specify a custom ODBC connection string for the source or target, the replication tasks rebuild the ODBC connection string for the second connection attempt by switching between the Oracle SID and SERVICE_NAME values.
    By default, the Extractor and Applier use the SERVICE_NAME value to build the ODBC connection string for the first connection attempt. If you select the **Use SID instead of SERVICE_NAME** option on the **Source Database** or **Target Database** tab, the Extractor or Applier task uses the Oracle SID to build the ODBC connection string for the first connection attempt. If the first connection attempt fails, the Extractor or Applier task switches to the other value and builds the ODBC connection string based on this value for the second connection attempt.
  - If the replication configuration specifies a custom ODBC connection string for the source or target, the replication tasks build the ODBC connection string for the second connection attempt based on the connection settings or database connection that you specify on the **Source Database** or **Target Database** tab.

Default value: 1

global.file_open_timeout

Specifies the number of seconds that a Data Replication component retries opening a file after the first open attempt fails. If this timeout period elapses without the file being opened, the component ends with an error message. This parameter affects only Data Replication components that run on Linux.

Valid values are 1 to 2147483647.

Default value: 600

global.fix_invalid_db_characters

For Amazon Redshift, Greenplum, and Teradata targets, indicates whether to fix UTF-8 encoded characters in character data that the targets cannot process correctly during InitialSync and Applier processing.

Note: The UTF-8 encoded characters are valid but the targets do not support them.

Options are:

- **0**. Do not fix UTF-8 encoded symbols in the character data that the target cannot process correctly. The InitialSync and Applier tasks might fail.
- **1**. Replace each byte of UTF-8 encoded symbols in character data that the target cannot process correctly with the character that is specified in global.invalid_character_replacement parameter.

Default value: 0

global.fix_invalid_encoding_characters

For Amazon Redshift, Greenplum, Netezza, Teradata, and Vertica targets, indicates whether to fix invalid UTF-8 byte sequences in character data during InitialSync and Applier processing. Options are:

- **0**. Do not fix invalid UTF-8 byte sequences. The InitialSync and Applier tasks might fail.
- **1**. If the global.icu_enabled parameter is set to 1 or the source data contains UTF-16 characters, replace invalid UTF-8 byte sequences with the character that is specified in the global.invalid_character_replacement parameter. If the global.icu_enabled parameter is set to 0 or the source data contains only UTF-8 characters, replace each byte of the invalid UTF-8 byte sequences with the character that is specified in the global.invalid_character_replacement parameter.

Default value: 0

global.handle_os_signals

For Applier, Extractor, and InitialSync tasks that run on Linux or UNIX, indicates which error log files these tasks generate in the *DataReplication_installation* directory when they end abnormally. Options are:

- **0**. Generate a core file.
- **1**. Generate a dbsync_crash.log file.
- **2**. Generate a core file and dbsync_crash.log file.

Default value: 2

Important: Change this value only at the request of Informatica Global Customer Support.

global.icu_enabled

For configurations that have a DB2 for Linux, UNIX, and Windows, Microsoft SQL Server, or Oracle source and an Amazon Redshift, Greenplum, Netezza, Vertica, or Teradata target, indicates whether the Applier and InitialSync convert character data from the source database encoding to the target database encoding. Options are:

- **0**. Do not convert the source character data to the target database encoding. Use this setting if the source and target databases use incompatible character sets and the source character data includes only Latin characters.

  Important: If the source character data includes non-Latin characters and the source and target databases use incompatible character sets, Data Replication ends with an error regardless of the global.icu_enabled setting.
- **1**. Convert the source character data to the target database encoding.

Default value: 0

Note: The Data Replication Console sets this parameter to 1 for new configurations.

global.insert_buffer_size

Specifies the maximum number of DML operations that the Applier and InitialSync accumulate in the internal buffer prior to flushing the buffer contents to the target. Each buffer stores operations of a single type.

If you use the default value of 0, the Applier and InitialSync determine the appropriate number of operations that can be buffered for the target type. The Applier and InitialSync use one of the following maximums:

- 1 operation for MySQL and PostgreSQL targets.
- 100 operations for Teradata targets.
- 200 operations for DB2 for Linux, UNIX, and Windows, Greenplum, Microsoft SQL Server, Netezza, Vertica, and Oracle targets.

If a replication configuration includes mapped LOB columns, Data Replication always uses the parameter value of 1.

Note: For MySQL and PostgreSQL targets, enter a value other than the default value only at the request of Informatica Global Customer Support.

Set this parameter to 1 to ensure correct Applier processing in the following cases:

- You configure Data Replication to continue Applier processing when specific database errors occur.

In other cases, use the default value of 0 for optimal Applier performance. If you set this parameter to 1 or any other low non-zero value, Applier performance might be degraded.

Default value: 0

global.invalid_character_replacement

Specifies the ASCII character that replaces UTF-8 encoded characters during InitialSync and Applier processing in the following situations:

- If you set the global.fix_invalid _db_characters parameter to 1 for Greenplum and Teradata targets, replaces UTF-8 encoded characters that the targets cannot process correctly.
- If you set the global.fix_invalid_encoding_characters parameter to 1 for Amazon Redshift, Greenplum, Netezza, Teradata, and Vertica targets, replaces invalid UTF-8 byte sequences.

Default value: ? (question mark)

global.lob_truncation_size

Determines the maximum size of a source LOB value, in bytes, that the Applier and InitialSync replicate to the target. If the size of a LOB value exceeds this limit, the data is truncated to the specified maximum size.

Default value: 0

If you use the default value of 0, the Applier and InitialSync truncate the source LOB data to 16 KB for Merge Apply targets and to 16 MB for other targets.

Notes:

- Netezza supports row sizes up to 64 KB when the data is loaded with Netezza high-performance loaders.
- The Extractor does not extract more than 50 MB from source LOB columns. Consequently, the maximum supported LOB size for change data capture is 50 MB, regardless of this parameter setting.
- If a target database uses a multibyte encoding, the Applier might truncate the last character from a character LOB column, causing an incomplete byte sequence. The Applier cannot apply the incomplete byte sequence.

  To avoid this problem on Amazon Redshift, Greenplum, Netezza, Teradata, and Vertica targets, set the global.fix_invalid_db_characters and global.fix_invalid_encoding_characters runtime parameters to 1. The Applier then replaces each byte of the invalid UTF-8 byte sequence or the entire sequence with the character that is specified in the global.invalid_character_replacement parameter.

global.max_output_log_size

Specifies the maximum size of a log file, in megabytes, to which an Extractor, Applier, or InitialSync task logs messages. When this log size limit is exceeded, the Data Replication task creates a new log file.

Note: If the time period in the global.time_for_output_log_rotation parameter elapses before this log size limit is exceeded, the Data Replication task creates a new log file.

Default value: 10

global.number_of_historical_metadata_records_kept

The maximum number of records that represent processed intermediate files in the Applier SQLite database. When the number of intermediate file records in the Applier SQLite database exceeds this value, the Applier deletes records for the earliest intermediate files.

Default value: 40000 records.

global.nz_pg_empty_strings_as_null

For Greenplum, Vertica, PostgreSQL, and Netezza targets, indicates whether the Applier and InitialSync load empty strings and LOBs from the source as null values on the target. Options are:

- **0**. Load empty strings and LOBs from the source as empty strings on the target.
- **1**. Load empty strings and LOBs from the source as null values on the target.

Default value: 1

global.print_runtime_settings

Indicates whether the InitialSync, Extractor, and Applier tasks log a list of the runtime parameters for which you specified a non-default value to the execution logs each time the tasks start. Options are:

- **0**. Do not log the list of parameters with non-default values.
- **1**. Log the list of parameters with non-default values.

Default value: 1

global.print_timestamps

Indicates whether to append date and time to the logged messages that provide debug information. Options are:

- **0**. Do not append the date and time.
- **1**. Append the date and time.

Default value: 1

global.show_transaction_locking

Indicates whether the Applier and InitialSync tasks retrieve information about database locking sessions that other applications establish. Options are:

- **0**. The Applier and InitialSync tasks do not provide information about database locking sessions and silently wait until the sessions release the row lock to continue processing.
- **1**. If the Applier or InitialSync task cannot load data into a locked target row, the task retrieves information about the database locking sessions and includes this information in the processing statistics. The Applier updates processing statistics at 5-minute intervals by default. InitialSync updates processing statistics at 10-second intervals by default. To change the default interval for the Applier or InitialSync task, use the apply.print_statistics_interval or initial.print_statistics_interval runtime parameter.

Default value: 1

global.time_for_output_log_rotation

Specifies the maximum number of hours that an Extractor, Applier, or InitialSync task can write messages to its task execution log. When this time period elapses, the task creates a new log file.

Note: If the size of a log file exceeds the global.max_output_log_size parameter value before this time period elapses, the Data Replication task creates a new log file.

Default value: 24

initial.check_empty_tables

Indicates whether InitialSync checks mapped target tables to determine if they are empty before loading data. Options are:

- **0**. Do not check if the target tables are empty. InitialSync loads data to all of the mapped tables on the target.
- **1**. Check if the target tables are empty. If InitialSync detects that one or more tables are not empty, it skips these tables and continues to load data to other tables. If InitialSync successfully loads data to all of the empty target tables, it ends without error.
- **2**. Check if the target tables are empty. If InitialSync detects that one or more target tables are not empty, it skips these tables and continues to load data to other tables. After InitialSync processes all of the target tables, it returns error code 57.

For configurations with multiple sources and a single target, set this parameter to 0 to materialize the target from all of the sources.

Important: For Oracle sources, disable the DBLinks option.

Default value: 0

initial.enforce_odbc_load

Indicates whether InitialSync uses a native database load utility or the appropriate ODBC driver to load data to the target. Options are:

- **0**. Use a native database load utility to load data to the target, if a load utility is available for the target type.
- **1**. Use the appropriate ODBC driver or the Oracle Call Interface (OCI) library to load data to the target. If you defined Tcl or SQL expressions that the target native load utility does not support, select this option so that InitialSync can properly process the expressions.
  InitialSync does not support the use of ODBC drivers for loading Vertica and Netezza targets.

Default value: 0

initial.lob_prefetch_threshold

For source tables that contain LOB columns, determines the number of rows that InitialSync gets in a single fetch. Large values provide better InitialSync performance but require a lot of memory to allocate prefetched rows. Use the following formula to estimate the allocated memory size:

```
allocated_memory_size = row_size * number_of_rows
```

Default value: 0

If you use the default value of 0, InitialSync gets the following number of rows in a single fetch:

- 1000 rows for Oracle sources
- 1 row for DB2 for Linux, UNIX, and Windows, Microsoft SQL Server sources

Note: For source tables that do not contain LOB columns, use the initial.select_buffer_size runtime parameter.

initial.named_transactions

For DB2, Microsoft SQL Server and Oracle targets, controls whether InitialSync uses a specific transaction or application name for the transactions that load data to targets during bidirectional or cascade replication. The transaction or application name can be the default name of DbSyncTransaction or an override name that you specify in the TX_NAME command line parameter. Options are:

- **0**. InitialSync does not use a specific transaction or application name for the transactions that insert data into targets.
- **1**. InitialSync uses one of the following transaction or application names:
  - For Microsoft SQL Server and Oracle targets, the default transaction name of DbSyncTransaction.
  - For DB2 targets, the default application name of DbSyncTransaction.
  - For any of these target types, an override transaction or application name that you specify with the TX_NAME command line parameter.

Default value: 1

initial.pipe_directory

Specifies the path and directory where pipes (*nix) are created when InitialSync runs. On Windows, you cannot specify this directory because pipes are created as `\\.\pipe\PipeName`. Informatica recommends that you specify this parameter for DB2 for Linux, UNIX, and Windows sources.

Default value: The Applier output directory

initial.pipe_wait_time

For DB2, Greenplum, PostgreSQL, Vertica and Netezza targets, specifies the number of seconds that InitialSync waits for the database native load utility to connect to the pipe that InitialSync opens for loading data to the target.

Increase this parameter value if you get the following error:

```
Write to pipe timed out write_size from bytes_to_write by thread thread_name to
the file file_name
```

Default value: 20 seconds on Linux and UNIX, 5 seconds on Windows

initial.print_statistics_interval

For configurations with targets that use ODBC drivers, specifies the number of seconds that must elapse before Data Replication updates InitialSync processing statistics in the InitialSync output log. The statistics include the number of processed rows, the total number of rows to be processed, and the estimated time for processing to complete. If you specify 0, Data Replication does not include this information in the InitialSync output log.

Default value: 10 seconds

initial.rows_to_commit

Specifies the maximum number of Insert rows that a single InitialSync thread can commit on the target. Use this parameter to avoid large transactions that might degrade load performance. If you use the ODBC drivers to connect to targets, this parameter applies to all target database types. If you use the target native load utilities, this parameter applies to the following target types:

- Greenplum
- Microsoft SQL Server
- PostgreSQL
- Vertica

Default value: 2147483647 rows

initial.script_engine_enabled

Indicates whether to enable or disable Tcl scripts and SQL expressions during InitialSync synchronization of source and target tables. Options are:

- **0**. Disable Tcl scripts and SQL expressions during InitialSync synchronization. The InitialSync task does not include target columns that are mapped to virtual columns with Tcl scripts or SQL expressions in the Insert statements for loading data.
- **1**. Enable Tcl scripts and SQL expressions during InitialSync synchronization.

Default value: 1

initial.select_buffer_size

Specifies the buffer size, in rows, that InitialSync uses for selecting data from source tables that do not contain mapped LOB columns. For source tables that contain mapped LOB columns, use the initial.lob_prefetch_threshold runtime parameter.

Default value: 0

If you use the default value of 0, InitialSync uses one of the following buffer sizes:

- 100 rows for Microsoft SQL Server and Oracle sources

- 100 rows for DB2 sources on Windows
- 1 row for DB2 sources on Linux and UNIX

Note: Values greater than 200 might enhance InitialSync performance but cause higher memory consumption.

initial.verbose_level

For Oracle sources, if you use InitialSync subtask threads, specifies the verbosity level for IniitalSync output. Options are:

- **0**. Provides standard InitialSync output.
- **1**. Provides additional output for each subtask thread.

For sources other than Oracle, use the default value of 0.

Default value: 0

initialsync.abort_on_failed_table

Indicates whether the InitialSync task ends abnormally if an error occurs during initial synchronization of source and target tables. Options are:

- **0**. InitialSync skips processing the target table when an error occurs but continues processing the other tables.
- **1**. InitialSync ends with an error and does not synchronize the remaining tables.

Default value: 1

initialsync.progress_print_step

Specifies the number of rows that the Extractor, Applier, or InitialSync component must process before Data Replication updates the progress information that is displayed when you start the Extractor or InitialSync component from the Data Replication Console.

Default value: 10000

pgloader.exceptions_path

Specifies the location of the log file that contains exceptions for a failed data load operation during an InitialSync or Applier job.

Default value: The Applier output directory

pgloader.reject_max

Specifies the maximum number of rows that can fail to be loaded before the Applier or InitialSync task fails.

Default value: 0

pgloader.reject_data_path

Specifies the location of the file that contains the data that failed to be loaded during an InitialSync or Applier task.

Default value: The Applier output directory

DDL Replication Parameters

apply.skip_add_column_failed_ddl

Determines the Applier behavior when it encounters an error replicating an ADD COLUMN operation. Options are:

- **0**. Issue an error and stop the Applier processing.

- **1**. Skip the failed operation and continue the Applier processing.

If the apply.skip_all_failed_ddl parameter is set to 1, the Applier skips the failed operation and continues processing, regardless of this parameter setting.

Default value: 0

apply.skip_all_failed_ddl

Determines the Applier behavior when it encounters an error replicating any DDL operation. Options are:

- **0**. Issue an error and stop the Applier processing.
- **1**. Skip the failed operation and continue the Applier processing.

Default value: 0

apply.skip_alter_column_failed_ddl

Determines the Applier behavior when it encounters an error replicating an ALTER COLUMN or MODIFY COLUMN operation. Options are:

- **0**. Issue an error and stop the Applier processing.
- **1**. Skip the failed operation and continue the Applier processing.

If the apply.skip_all_failed_ddl parameter is set to 1, the Applier skips the failed operation and continues processing, regardless of this parameter setting.

Default value: 0

apply.skip_create_index_failed_ddl

Determines the Applier behavior when it encounters an error replicating an ADD INDEX or CREATE INDEX operation. Options are:

- **0**. Issue an error and stop the Applier processing.
- **1**. Skip the failed operation and continue the Applier processing.

If the apply.skip_all_failed_ddl parameter is set to 1, the Applier skips the failed operation and continues processing, regardless of this parameter setting.

Default value: 0

apply.skip_create_table_failed_ddl

Determines the Applier behavior when it encounters an error replicating a CREATE TABLE operation. Options are:

- **0**. Issue an error and stop the Applier processing.
- **1**. Skip the failed operation and continue the Applier processing.

If the apply.skip_all_failed_ddl parameter is set to 1, the Applier skips the failed operation and continues processing, regardless of this parameter setting.

Default value: 0

apply.skip_drop_column_failed_ddl

Determines the Applier behavior when it encounters an error replicating a DROP COLUMN operation. Options are:

- **0**. Issue an error and stop the Applier processing.
- **1**. Skip the failed operation and continue the Applier processing.

If the apply.skip_all_failed_ddl parameter is set to 1, the Applier skips the failed operation and continues processing, regardless of this parameter setting.

Default value: 0

apply.skip_drop_index_failed_ddl

Determines the Applier behavior when it encounters an error replicating a DROP INDEX operation. Options are:

- **0**. Issue an error and stop the Applier processing.
- **1**. Skip the failed operation and continue the Applier processing.

If the apply.skip_all_failed_ddl parameter is set to 1, the Applier skips the failed operation and continues processing, regardless of this parameter setting.

Default value: 0

apply.skip_drop_table_failed_ddl

Determines the Applier behavior when it encounters an error replicating a DROP TABLE operation. Options are:

- **0**. Issue an error and stop the Applier processing.
- **1**. Skip the failed operation and continue the Applier processing.

If the apply.skip_all_failed_ddl parameter is set to 1, the Applier skips the failed operation and continues processing, regardless of this parameter setting.

Default value: 0

Applier Thread Distribution Parameters

apply.distribute_by_obj_size_threshold

If you enable subtask Applier threads by setting the apply.subtasks_enabled parameter to 1, this parameter determines the threshold load amount, in bytes, for the subtask Applier threads. When this threshold is reached, another subtask thread is created. The default value of 0 sets the threshold at the lowest subtask thread load.

Typically, use the default value because it allows Data Replication to calculate the threshold dynamically. However, you might need to set this value manually in some situations. For example, if a target table has very few changes and the other target tables have many changes, you might need to set the threshold to a value larger than the lowest load on the subtask thread for the table with few changes to prevent spawning multiple subtask Applier threads.

Default value: 0

apply.global_transaction_support

Indicates whether to use global transactions when applying changes to Oracle targets. Options are:

- **0**. Disable global transactions and commit data on each subtask thread or on each Applier thread if you do not use subtasks.
- **1**. Enable global transactions and perform a single commit for all Applier threads or subtask threads at the end of the apply cycle. This option might degrade performance.
- **2**. Enable global transactions and perform a commit on each Applier thread.

Default value: 1

apply.max_subtasks

If you set the apply.subtasks_enabled parameter to 1 to enable creation of subtask threads for each Applier task, this parameter specifies the maximum number of subtask threads that can be created. Ensure that the number of apply subtask threads is not larger than the number of target database connections.

Default value: 4

apply.subtasks_enabled

Indicates whether Data Replication can create subtask threads for each Applier task thread. The Applier can create subtasks only when loading change data to target tables in SQL Apply mode. If you enable subtask threads, Data Replication distributes changes across threads by object rather than by row. Options are:

- **0**. Do not use subtask threads.
- **1**. Use subtask threads. Select this option to provide more parallelism in Applier processing in SQL Apply mode. For example, use this option if you have few tables to which to apply many changes. In this case, enabling subtask threads for each Applier task results in better performance because Data Replication uses more threads to process the tables.

Note: Multiple subtask threads can reduce available CPU resources for other applications.

Default value: 0

APPENDIX D

Command Line Parameters for Data Replication Components

This appendix includes the following topics:

About Command Line Parameters

You can start the InitialSync, Extractor, Applier, and Server Manager executables or scripts from the command line with optional parameters.

To enter the parameters at the command line, use the following syntax:

- On Windows:
  DataReplication_component.exe [*parameter1 parameter2* ...]
- On Linux or UNIX:
  ./DataReplication_component [*parameter1 parameter2* ...]

The ellipsis (...) indicates that additional parameters can follow.

The parameter descriptions use the following conventions:

- *Italics* indicate a variable.
- Curly brackets { } indicate that you must enter one of the options for the parameter.
- Underlining indicates the default option.

Related Topics:

- "Executables Called from the Data Replication Console or Scripts" on page 390

Command Line Parameters for InitialSync

You can start the InitialSync executable or script from the command line with the following optional parameters:

CLEAN_SQLITEREP= {y|n}

Removes repository information about previous revisions to the current configuration file from the revision control system. Options are:

- **y**. Remove the repository information.
- **n**. Do not remove the repository information.

Default value: n

CONFIG=*path/configuration_SQLite_database_file*

Specifies the path and file name of a configuration SQLite database file for a replication configuration.

Default value: No default

CONN_ID=*target_connection_ID*

Specifies the connection ID of the connection to the target database that is specified in the replication configuration.

Default value: No default

DBLINK={y|n}

For Oracle-to-Oracle replication only, indicates whether Data Replication can use dblinks to perform initial synchronization. The use of dblinks can improve performance. Options are:

- **y**. Enables Data Replication to use dblinks.
- **n**. Prevents Data Replication from using dblinks.

Default value: y

DEST_TABLES=*table1,table2,…*

DEST_TABLES=*table_mask*

Lists the target tables to resynchronize. If you set the RESYNC parameter to **n**, use this parameter to selectively resynchronize the target tables.

Note: You can include the question mark (?) and asterisk (*) wildcards to define a mask for the tables to select for resynchronization.

Default value: By default, InitialSync includes all tables that require resynchronization.

DIRECT={y|n}

Indicates whether InitialSync uses a native database load utility or the appropriate ODBC driver to load data to the target. Options are:

- **y**. Use a native database load utility to load data to the target, if a load utility is available for the target type.

- **n**. Use the appropriate ODBC driver or the Oracle Call Interface (OCI) library to load data to the target. InitialSync does not support the use of ODBC drivers for loading Vertica and Netezza targets.

Default value: n for Oracle targets, y for other targets

EXCLUDE_DEST_ TABLES=*table1, table2,…*

EXCLUDE_DEST_ TABLES=*table_mask*

Lists the target tables to exclude from resynchronization. If you set the RESYNC parameter to **n**, use this parameter to selectively exclude target tables from the resynchronization.

Note: You can use the question mark (?) and asterisk (*) wildcards to specify a mask for the tables to exclude.

Default value: By default, InitialSync includes all tables for resynchronization.

JOURNAL_MODE={DELETE|TRUNCATE|PERSIST|MEMORY|WAL|OFF}

Specifies the rollback journal mode for the SQLite repository that contains configuration information. Journal mode options are:

- **DELETE**. Delete the rollback journal at the conclusion of each transaction.
- **TRUNCATE**. Commit transactions by truncating the rollback journal to zero-length instead of deleting it.
- **PERSIST**. Prevent the rollback journal from being deleted at the end of each transaction.
- **MEMORY**. Store the rollback journal in volatile RAM.
- **WAL**. Use a write-ahead log instead of a rollback journal to implement transactions.
- **OFF**. Disable the rollback journal completely.

For more information about SQLite journal modes, see the SQLite documentation.

Default value: DELETE for Linux and UNIX, WAL for Windows

Ensure that this parameter has the same value for all Data Replication components.

INSERT_ARRAY_ SIZE=*size*

Specifies the number of rows in the buffer when Data Replication uses ODBC connectivity to perform Inserts on the target.

Default value: 100

NON_EMPTY_DEST= {y|n}

For Teradata targets, specifies whether data is appended to target tables or written to empty tables. Options are:

- **y**. Write data to an empty Teradata target.
- **n**. Append data to an existing Teradata target.

Default value: n

OUTPUT_LOG_NAME=*file_name*

Specifies the name of the first execution log file that contains output from the Data Replication component.

Default value: No default

OUTPUT_LOGS_FOLDER=*path_to_directory*

Specifies the directory in which Data Replication creates execution log files for the Data Replication component.

Default value: No default

RESYNC={y|n|p}

Indicates whether InitialSync synchronizes all of the mapped target tables or a subset of the mapped target tables. Options are:

- **y**. Synchronize all of the mapped target tables with the source tables.
- **n**. Synchronize the set of tables that you specify in the DEST_TABLES parameter. You can optionally use the EXCLUDE_DEST_TABLES parameter to exclude some of these tables.
- **p**. Synchronize the mapped target tables for which Sync Point value is -1.

Default value: y

SM_RUN_ID=*server_manager_run_ID*

Specifies the run ID that the Server Manager assigns to the replication task.

Important: The Server Manager uses the SM_RUN_ID parameter to set the run ID value when starting a task. Do not change this parameter value or delete this parameter from the commands that start replication tasks.

Default value: No default

SOURCE_DATA_IS_READ_ONLY={y|n}

For Oracle sources, indicates whether the InitialSync task uses a regular SELECT query or an Oracle Flashback query to unload data from the source. Options are:

- **n**. Run a Flashback query with a SELECT statement that includes an AS OF SCN clause. This query returns committed data that was current at the point in time that corresponds to a particular SCN.
- **y**. Run a regular SELECT query. Use this option to unload data from an Oracle standby database that is open for read-only access and does not support Flashback mode.

  If you run the SELECT statement with AS OF SCN clause for an Oracle standby database that is open for read-only access, Oracle issues the following error message:

  ```
  ORA-08183: Flashback cannot be enabled in the middle of a transaction
  ```

Default value: n

SQLITE_CACHE_SIZE=*cache_size*

Specifies the maximum number of database disk pages that SQLite will hold in memory at once per open database file. The Data Replication component uses this parameter to open SQLite database files.

Default value: 16384 bytes

Change the default value only at the request of Informatica Global Customer Support.

SQLITE_PAGE_SIZE=*page_size*

Specifies the page size of the SQLite database when a new database is created, a database is still empty, or prior to running the VACUUM command. The Data Replication component uses this parameter to open SQLite database files.

Default value: 16384 bytes

Change the default value only at the request of Informatica Global Customer Support.

THREADS=*number_of_threads*

Specifies the number of threads for the InitialSync component. You can use this parameter to override the value that is specified in the configuration file.

Default value: No default

TX_NAME=*transaction_name*

For Microsoft SQL Server, Oracle targets, specifies the transaction name for the transactions that the Applier and InitialSync use to replicate data. For DB2 targets, specifies the application name that the Applier and InitialSync use to replicate data. Use this parameter to override the default name of DbSyncTransaction.

Important: Ensure that the transaction name or application name is valid for the target database. For more information about valid identifiers, see your target database documentation.

Default value: DbSyncTransaction

UDP_SERVER_PORT=*port_number*

Specifies the User Datagram Protocol (UDP) port of the Server Manager through which the Data Replication component communicates with the Server Manager. Valid values are integers from 1 through 65535.

Default value: 8087

Important: Ensure that the UDP_SERVER_PORT parameter value matches the value of the UDP_PORT parameter for the Server Manager.

Command Line Parameters for the Extractor

You can start the Extractor executable or script from the command line with the following optional parameters:

CLEAN_SQLITEREP= {y|n}

Removes repository information about previous revisions to the current configuration file from the revision control system. Options are:

- **y**. Remove the repository information.
- **n**. Do not remove the repository information.

Default value: n

CONFIG=*path/configuration_SQLite_database_file*

Specifies the path and file name of a configuration SQLite database file for a replication configuration.

Default value: No default

FORCE_EXTRACT_ REGARDLESS_SYNC_ SCN={y|n}

For DB2 for Linux, UNIX, and Windows sources, indicates whether to extract changes from source tables that appear to not have been previously synchronized with the target. For these tables, the previous Extractor run did not write the Sync Point (LSN) values in the DBSYNC_SYNC_LSN to the configuration file. Options are:

- **y**. Force change data extraction for source tables that do not have a Sync Point value in the configuration file.
- **n**. Do not extract change data for source tables that do not have a Sync Point value in the configuration file.

Default value: n

JOURNAL_MODE={DELETE|TRUNCATE|PERSIST|MEMORY|WAL|OFF}

Specifies the rollback journal mode for the SQLite repository that contains configuration information. Journal mode options are:

- **DELETE**. Delete the rollback journal at the conclusion of each transaction.

- **TRUNCATE**. Commit transactions by truncating the rollback journal to zero-length instead of deleting it.
- **PERSIST**. Prevent the rollback journal from being deleted at the end of each transaction.
- **MEMORY**. Store the rollback journal in volatile RAM.
- **WAL**. Use a write-ahead log instead of a rollback journal to implement transactions.
- **OFF**. Disable the rollback journal completely.

For more information about SQLite journal modes, see the SQLite documentation.

Default value: DELETE for Linux and UNIX, WAL for Windows

Ensure that this parameter has the same value for all Data Replication components.

INT_FILES_DIR=*path/directory_name*

Specifies the directory that contains the intermediate files. The Extractor transmits intermediate files to the specified directory. The Applier gets intermediate files from the specified directory.

Default value: *DataReplication_installation*/output

KEYS_STORAGE_PATH

Specifies the path to the internal master key that the Oracle Extractor uses to decrypt Oracle keys from a replication configuration. When the Server Manager starts the Extractor, it sets this command-line parameter to the **Internal Master Key Path** value that you set in the Server Manager advanced properties in the Data Replication Console.

ONLINE_FILES_PREFIX=*path_to_the_mounted_directory*

For Oracle sources, specifies the directory on the local file system that the Extractor searches for online redo logs. By default, the Extractor searches for online redo logs in the location that it gets from the Oracle server. Use this parameter to extract change data from online redo logs over the network from the mounted file system.

For example, if Oracle uses the redo online log /u01/app/oracle/oradata/orcl/redo01.log, and you mounted the directory with online redo logs to the directory C:\APP\USERNAME\ORADATA\ORCL, use the following command to run the Extractor:

```
extract.cmd some_configuration.xml ONLINE_FILES_PREFIX=C:\APP\USERNAME\ORADATA
\ORCL\
```

The Extractor can access the online redo log C:\APP\USERNAME\ORADATA\ORCL\redo01.log.

Default value: No default

OUTPUT_LOG_NAME=*file_name*

Specifies the name of the first execution log file that contains output from the Data Replication component.

Default value: No default

OUTPUT_LOGS_FOLDER=*path_to_directory*

Specifies the directory in which Data Replication creates execution log files for the Data Replication component.

Default value: No default

REALTIME_ MODE={y|n}

Enables continuous replication. Options are:

- **y**. Run the Extractor or Applier in continuous replication mode.
- **n**. Run the Extractor or Applier on demand.

Default value: n

SKIP_CONTINUITY_CHECK={y|n}

For Microsoft SQL Server and Oracle sources, indicates whether to skip integrity checking for the database logs during the first Extractor run.

- **y**. Skip integrity checking for the database logs during the first Extractor run.
- **n**. Perform integrity checking for the database logs. The start time of each log must match the end time of the previous log record. The Extractor stops processing logs when it detects any inconsistency.

Default value: n

SKIP_DEFAULT_TX={y|n}

Indicates whether to capture change data for transactions that have the default name of DbSyncTransaction. Options are:

- **y**. Skip change data capture for the transactions that have the name of DbSyncTransaction.
- **n**. Capture change data for the transactions that have the name of DbSyncTransaction.

Default value: y

SKIP_TEM_ISSUES_FOR_THREADS=*redo_thread1,redo_thread2, ...*

For Oracle RAC sources, specifies the list of redo threads for which the Extractor ignores Thread Enable Marker (TEM) issues. If a TEM issue occurs, the Extractor continues processing redo log records. The following types of TEM issues indicate that the Extractor might have missed change data records for a redo thread:

- The Extractor encounters a TEM for an online Oracle redo thread.
- A TEM for an offline redo thread refers to an unexpected log sequence number.

Ignoring these TEM issues for a redo thread might result in change data loss.

Use this parameter only if you must continue the Extractor processing despite possible change data loss.

Default value: None

SKIP_TX=*transaction_names*

Specifies a comma-separated list of transaction names to skip. When extracting change data, the Extractor skips all of the transactions that have matching names. You can use the asterisk (*) and question mark (?) wildcard characters to specify a mask for the transaction names. For example, use the following value to skip all of the transactions that have the name "A_Transaction" and all of the transactions that have the name starting with "B":

```
SKIP_TX=A_Transaction,B*
```

Default value: No default

SM_RUN_ID=*server_manager_run_ID*

Specifies the run ID that the Server Manager assigns to the replication task.

Important: The Server Manager uses the SM_RUN_ID parameter to set the run ID value when starting a task. Do not change this parameter value or delete this parameter from the commands that start replication tasks.

Default value: No default

SQLITE_CACHE_SIZE=*cache_size*

Specifies the maximum number of database disk pages that SQLite will hold in memory at once per open database file. The Data Replication component uses this parameter to open SQLite database files.

Default value: 16384 bytes

Change the default value only at the request of Informatica Global Customer Support.

SQLITE_PAGE_SIZE=*page_size*

Specifies the page size of the SQLite database when a new database is created, a database is still empty, or prior to running the VACUUM command. The Data Replication component uses this parameter to open SQLite database files.

Default value: 16384 bytes

Change the default value only at the request of Informatica Global Customer Support.

UDP_SERVER_PORT=*port_number*

Specifies the User Datagram Protocol (UDP) port of the Server Manager through which the Data Replication component communicates with the Server Manager. Valid values are integers from 1 through 65535.

Default value: 8087

Important: Ensure that the UDP_SERVER_PORT parameter value matches the value of the UDP_PORT parameter for the Server Manager.

Command Line Parameters for the Applier

You can start the Applier executable or script from the command line with the following optional parameters:

APPLY_THREADS=*number_of_threads*

Specifies the number of threads to use during apply processing. Maximum is 256. Use this parameter to override the value that is specified in the configuration file.

Default value: No default

CLEAN_SQLITEREP= {y|n}

Removes repository information about previous revisions to the current configuration file from the revision control system. Options are:

- **y**. Remove the repository information.
- **n**. Do not remove the repository information.

Default value: n

CONFIG=*path/configuration_SQLite_database_file*

Specifies the path and file name of a configuration SQLite database file for a replication configuration.

Default value: No default

CONFIG_LOADER=*path/Applier_SQLite_database_file*

Specifies the path and file name of the Applier SQLite database file.

Default value: No default

CONN_ID=*target_connection_ID*

Specifies the connection ID of the connection to the target database that is specified in the replication configuration.

Default value: No default

-DUMP

Causes the Applier to dump the contents of the intermediate files instead of applying the changes to the target. Place this argument immediately after the configuration file name. Use this parameter for debugging purposes.

Default value: Not applicable

FORCE={y|n}

Indicates whether to apply changes to a target that was not previously synchronized with the source. Options are:

- **y**. Force the Applier to apply changes even if the DB2 Extractor or InitialSync has not written a Sync Point value to the configuration file.
- **n**. Do not apply changes if the DB2 Extractor or InitialSync has not written a Sync Point value to the configuration file.

Default value: n

JOURNAL_MODE={DELETE|TRUNCATE|PERSIST|MEMORY|WAL|OFF}

Specifies the rollback journal mode for the SQLite repository that contains configuration information. Journal mode options are:

- **DELETE**. Delete the rollback journal at the conclusion of each transaction.
- **TRUNCATE**. Commit transactions by truncating the rollback journal to zero-length instead of deleting it.
- **PERSIST**. Prevent the rollback journal from being deleted at the end of each transaction.
- **MEMORY**. Store the rollback journal in volatile RAM.
- **WAL**. Use a write-ahead log instead of a rollback journal to implement transactions.
- **OFF**. Disable the rollback journal completely.

For more information about SQLite journal modes, see the SQLite documentation.

Default value: DELETE for Linux and UNIX, WAL for Windows

Ensure that this parameter has the same value for all Data Replication components.

INT_FILES_DIR=*path/directory_name*

Specifies the directory that contains the intermediate files. The Extractor transmits intermediate files to this directory, and the Applier gets intermediate files from this directory.

Default value: *DataReplication_installation*/output

OUTPUT_LOG_NAME=*file_name*

Specifies the name of the first execution log file that contains output from the Data Replication component.

Default value: No default

OUTPUT_LOGS_FOLDER=*path_to_directory*

Specifies the directory in which Data Replication creates execution log files for the Data Replication component.

Default value: No default

REALTIME_ MODE={y|n}

Enables continuous replication. Options are:

- **y**. Run the Extractor or Applier in continuous replication mode.
- **n**. Run the Extractor or Applier on demand.

Default value: n

REMOVE_PROCESSED_FILES={y|n}

Indicates whether to remove processed intermediate files. Options are:

- **y**. Remove processed intermediate files.
- **n**. Keep processed intermediate files. Use this option if you might need the processed intermediate files for debugging purposes.

Default value: y

Important: When the Server Manager starts the Applier, it always sets this parameter to n.

SM_RUN_ID=*server_manager_run_ID*

Specifies the run ID that the Server Manager assigns to the replication task.

Important: The Server Manager uses the SM_RUN_ID parameter to set the run ID value when starting a task. Do not change this parameter value or delete this parameter from the commands that start replication tasks.

Default value: No default

SQLITE_CACHE_SIZE=*cache_size*

Specifies the maximum number of database disk pages that SQLite will hold in memory at once per open database file. The Data Replication component uses this parameter to open SQLite database files.

Default value: 16384 bytes

Change the default value only at the request of Informatica Global Customer Support.

SQLITE_PAGE_SIZE=*page_size*

Specifies the page size of the SQLite database when a new database is created, a database is still empty, or prior to running the VACUUM command. The Data Replication component uses this parameter to open SQLite database files.

Default value: 16384 bytes

Change the default value only at the request of Informatica Global Customer Support.

TX_NAME=*transaction_name*

For Microsoft SQL Server, Oracle targets, specifies the transaction name for the transactions that the Applier and InitialSync use to replicate data. For DB2 targets, specifies the application name that the Applier and InitialSync use to replicate data. Use this parameter to override the default name of DbSyncTransaction.

Important: Ensure that the transaction name or application name is valid for the target database. For more information about valid identifiers, see your target database documentation.

Default value: DbSyncTransaction

UDP_SERVER_PORT=*port_number*

Specifies the User Datagram Protocol (UDP) port of the Server Manager through which the Data Replication component communicates with the Server Manager. Valid values are integers from 1 through 65535.

Default value: 8087

Important: Ensure that the UDP_SERVER_PORT parameter value matches the value of the UDP_PORT parameter for the Server Manager.

Command Line Parameters for the Server Manager

You can run the Server Manager executable or script from the command line with the following optional parameters:

CLEAN

Deletes all data from the Server Manager SQLite repository, SM.db3, including server properties, connections, schedules, tasks, and other server settings.

Note: Do not use the clean parameter in combination with other parameters.

DBFILE="*file_name*"

Overrides the name of the internal SQLite repository file that the Server Manager uses for storing configuration file and internal service information.

DBSYNC_HOME=*path*

Overrides the DBSYNC_HOME environment variable value for a Server Manager instance. Use this option for debugging purposes or when you run multiple Server Manager instances from different DBSYNC_HOME locations on the same system.

DEBUG

Runs the Server Manager in debug mode. Use this parameter only at the request of Informatica Global Customer Support.

DESC="*service_description*"

If you run the Server Manager as a service on Windows, optionally specify a description for the Server Manager service. Also see the RUN_AS_SERVICE parameter.

HELP

Displays a list of Server Manager parameters on screen.

Note: Do not use the help parameter in combination with other parameters.

HTTP={0|1}

Indicates whether to use HTTP connections for the Server Manager. Options are:

- **0**. Do not use HTTP connections.
- **1**. Use HTTP connections.

If you do not specify the HTTP parameter or HTTPS parameter, when you start the Server Manager for the first time, it uses the default value of 1 to use HTTP connections. When you later start the Server Manager again, it uses the value from the preceding run.

Default value: 1

HTTPS={0|1}

Indicates whether to use HTTPS connections for the Server Manager. Options are:

- **0**. Do not use HTTPS connections.
- **1**. Use HTTPS connections.

If you do not specify the HTTPS parameter or HTTP parameter, when you start the Server Manager for the first time, it uses the default value of 1 for the HTTP parameter so that HTTP connections are used. When you later start the Server Manager again, it uses the value from the preceding run.

Default value: 0

HTTPS_PORT=*nnnn*

If you set the HTTPS parameter to 1, use this parameter to override the default listener port through which a client submits HTTPS requests to the Server Manager. Valid values are integers from 1 through 65535.

Default value: 8089

Important: If you run a Server Manager subserver later with a different port number, change the port number in the Data Replication Console on the **Server Manager** tab > **Servers** view.

INSTANCE="*server_name*"

If you run the Server Manager as a service on Windows or as a daemon on Linux or UNIX, optionally specify a name for the Server Manager instance that overrides the default name. Use this option if you run multiple Server Manager instances from different DBSYNC_HOME locations on the same system.

JOURNAL_MODE={DELETE|TRUNCATE|PERSIST|MEMORY|WAL|OFF}

Specifies the rollback journal mode for the SQLite repository that contains configuration information. Journal mode options are:

- **DELETE**. Deletes the rollback journal at the conclusion of each transaction.
- **TRUNCATE**. Commits transactions by truncating the rollback journal to zero length instead of deleting it.
- **PERSIST**. Prevents the rollback journal from being deleted at the end of each transaction.
- **MEMORY**. Stores the rollback journal in volatile RAM.
- **WAL**. Uses a write-ahead log instead of a rollback journal to implement transactions.
- **OFF**. Disables the rollback journal completely.

For more information about SQLite journal modes, see the SQLite documentation.

Default value: DELETE for Linux and UNIX, WAL for Windows

Ensure that this parameter has the same value for all Data Replication components.

PORT=*nnnn*

If the HTTP parameter is set to 1, overrides the default listener port through which a client submits HTTP requests to the Server Manager. Valid values are integers from 1 through 65535.

Default value: 8088

Important: If you run a Server Manager subserver later with a different port number, change the port number in the Data Replication Console on the **Server Manager** tab > **Servers** view.

RESET_IDRADMIN

Resets the password for the idradmin account to empty. This parameter also unlocks the idradmin account if the account is locked out after the maximum number of failed login attempts that is specified in the MaxLoginAttempts property is reached.

Note: Do not use the RESET_IDRADMIN parameter in combination with other Server Manager command-line parameters.

RUN_AS_SERVICE {-i|-r|-s|-k|-d}

On Windows, enter the RUN_AS_SERVICE parameter with one of the following options to install, start, or manage the Server Manager as a service:

- **-i**. Installs the server_manager program as a service but does not start it.

- **-r**. Removes the server_manager service from the Service Manager.
- **-s**. Starts the server_manager service.
- **-k**. Stops the server_manager service.
- **-d**. Runs the server_manager service in debug mode.

On Linux and UNIX, you do not need to use this parameter to start the daemon. The server_manager.sh script runs the Server Manager with the RUN_AS_SERVICE parameter by default.

START

On Linux and UNIX, starts the Server Manager instance.

Note: If you run the server_manager.sh script without the start parameter, the script still starts the Server Manager instance by default. However, Informatica recommends that you include the start parameter to clearly indicate intent.

STATE={0|1|2}

Determines whether to start a Server Manager instance as a Main server or a subserver. Options are:

- **0**. Start the Server Manager without a predefined role. Data Replication assigns a role to the Server Manager instance the first time you perform one of the following actions in the Data Replication Console: connect to the Server Manager or add it as a subserver.
- **1**. Start the Server Manager instance as the Main server. You will not be able to add this Server Manager instance as a subserver later from the Data Replication Console.
- **2**. Start the Server Manager as a subserver. You will not be able to connect to this Server Manager instance later from the Data Replication Console.

Default value: 0

STOP

On Linux and UNIX, stops the Server Manager instance.

UDP_PORT=*nnnn*

Overrides the default port through which the Server Manager communicates with the Data Replication components. Valid values are integers from 1 through 65535.

Default value: 8087

Examples:

The following command displays help information for Server Manager parameters:

```
server_manager.exe HELP
```

The following command removes the current Server Manager service:

```
server_manager.exe RUN_AS_SERVICE -r
```

APPENDIX E

Updating Configurations in the Replication Configuration CLI

This appendix includes the following topics:

Replication Configuration CLI Overview

The Replication Configuration Command Line Interface (CLI) is a command line utility that you can use to update source and target metadata for replication configurations that you created in the Data Replication Console and saved as an XML file.

You can use the Replication Configuration CLI to update an existing configuration from the command line. The CLI retrieves metadata from the source and target databases and updates source and target schema and object IDs in the configuration. The CLI can run in interactive or non-interactive mode.

For example, you might want to update the configuration from the command line when you run a script for deploying the configuration to another replication environment.

You can run the Replication Configuration CLI on a source or target system or on a standalone machine.

Before you start the Replication Configuration CLI the first time, verify that the prerequisites for the JDBC drivers are met. For more information, see the *Informatica Data Replication Installation and Upgrade Guide*.

To start the Replication Configuration CLI, run one of the following scripts, which are located in the *DataReplication_installation* directory:

- On Windows, run idr_cmd_line.cmd.
- On Linux or UNIX, run idr_cmd_line.sh.

Tip: In interactive mode, the Replication Configuration CLI maintains a history buffer of recently entered command statements. After you start the Replication Configuration CLI from the command line, use the Up Arrow and Down Arrow buttons to navigate through the history to a previously entered command statement.

Related Topics:

- "Editing a Replication Configuration" on page 340

Updating Source and Target Metadata for a Replication Configuration in the CLI

You can use the Replication Configuration Command Line Interface (CLI) to get updated source and target schema information and object IDs for a replication configuration.

Data Replication retrieves metadata from the source and target databases and updates the source and target schema and object IDs in the configuration.

You can get the updated metadata for the configuration in either of the following modes:

- In interactive mode, run the idr_cmd_line.cmd or idr_cmd_line.sh script without any parameters to display the Replication Configuration CLI prompt. At the prompt, enter the open command to open the XML configuration file, retrieve current source and target metadata, and update the configuration with that metadata. Then enter the save command to save the updated configuration.
- In non-interactive mode, run the idr_cmd_line.cmd or idr_cmd_line.sh script from the command prompt with the path to the configuration file as the first parameter. You can also include the following optional parameters:
  - Include the recovery parameter to specify the recovery table name if you did not previously create this table when generating the configuration or if you want to override the previously created table.
  - If you have DB2 sources, include the service parameter to specify the auxiliary table name if you did not previously create this table when generating the configuration or if you want to override the previously created table.

  The script retrieves the source and target metadata and updates the configuration.

Updating Source and Target Metadata for a Configuration in Interactive Mode

To update source and target metadata for a configuration in interactive mode, first use the open command with the /r option to open a XML configuration file and get the updated metadata. Then use the save command to save the updated configuration.

1. Enter the open command with the /r option to open the replication configuration for which you want to get updated source and target metadata.

   For example:

   ```
   open /r C:\configs\demo_config.xml
   ```

   Data Replication retrieves metadata from the source and target databases and updates source and target schema and object IDs in the configuration.
2. Enter the save command to save the updated configuration in XML format to the local file system.

   For example:

   ```
   save C:\configs\demo_config.xml
   ```

Updating Source and Target Metadata for a Configuration in Non-interactive Mode

To update source and target metadata for a configuration in non-interactive mode, enter the Replication Configuration CLI script at the prompt with the path to the XML configuration file as the first parameter. You can also include parameters that specify the recovery table name for the target and the auxiliary table name for a DB2 source.

1. In the *DataReplication_installation*/uiconf/default.cfg file, set the use_default_values_during_saving parameter to true.

   Note: If the default.cfg file does not exist, use a text editor to create the file in the uiconf directory and then add the parameter in the file. For more information, see "Default.cfg File" on page 391.

2. From the command line, run the Replication Configuration CLI script with the path to the configuration as the first positional parameter.

   - On Windows, use the following syntax:

     ```
     idr_cmd_line.cmd path_to_configuration [recovery=schema.recovery_table]
     [service=schema.auxiliary_table]
     ```

   - On Linux and UNIX, use the following syntax:

     ```
     idr_cmd_line.sh path_to_configuration [recovery=schema.recovery_table]
     [service=schema.auxiliary_table]
     ```

   Include the optional recovery and service parameters if you did not create the recovery table or auxiliary table for the configuration earlier or if you want to override the one that you previously created.

   - The recovery parameter specifies a recovery table on the target.
   - The service parameter specifies an auxiliary table on a DB2 source.

   Important: In the recovery and service parameters, the schema and table names are case-sensitive.

   The Replication Configuration CLI retrieves the source and target metadata and uses it to update the specified configuration.

   If the target tables that you mapped in the configuration do not have primary keys, the Replication Configuration CLI, by default, creates a virtual index for these tables and includes all table columns in the index. Alternatively, you can set the pk_warning_option parameter in the *DataReplication_installation*/uiconf/default.cfg file to 2 to disable replication of Update and Delete operations. With this parameter setting, Data Replication replicates only Insert operations.

   Note: If the default.cfg file does not exist, use a text editor to create the file in the uiconf directory and then add the pk_warning_option parameter in the file.

Related Topics:

- "Filtering Table Rows for Selective Replication to Different Targets" on page 301
- "Managing Open Transactions" on page 336

Replication Configuration CLI Commands

Use the following commands to work with configurations and display help information from the Replication Configuration Command Line Interface (CLI):

exit

Exits the Replication Configuration CLI.

help cmd

Displays a list of all of the Replication Configuration CLI commands.

help *command*

Displays a description of the specified command.

open /r *path_to_XML_configuration*

Opens the specified replication configuration XML file for updating.

save *path_to_XML_configuration*

Saves the replication configuration XML file to the specified path in the local file system.

APPENDIX F

DDL Statements for Manually Creating Recovery Tables

Data Replication requires recovery tables on most targets to prevent the Applier from replicating previously applied data again when the Applier task restarts.

The Applier uses recovery tables for all target types except Apache Kafka, Cloudera, Flat File, and Hortonworks. For Kafka targets, Data Replicaton uses a checkpoint file instead of a recovery table.

Typically, the Applier creates recovery tables. However, if an error occurs while the Applier is creating recovery tables, or if the recovery tables become damaged or corrupted, you can use the following CREATE TABLE statements to manually create the recovery tables.

In these DDL statements, replace the following strings with actual values:

- *db* represents the name of the target database with the recovery table.
- *owner* represents the name of the owner of the recovery table.
- *table* represents the table name of the recovery table.

Amazon Redshift, DB2 for Linux, UNIX, and Windows, Greenplum, Netezza , PostgreSQL, and Teradata:

```
CREATE TABLE "owner"."table" (config VARCHAR(256) NOT NULL, task_id INT NOT NULL,
subtask_id INT, scn DECIMAL(20,0) NOT NULL, scn_low DECIMAL(20,0), cycle_id INT,
table_name VARCHAR(256), status VARCHAR(256));
```

Microsoft SQL Server:

```
CREATE TABLE [db].[schema].[table] (config VARCHAR(256) NOT NULL, task_id INT NOT NULL,
subtask_id INT, scn DECIMAL(20, 0) NOT NULL, scn_low DECIMAL(20, 0), cycle_id INT,
table_name VARCHAR(256), status VARCHAR(256));
```

MySQL:

```
CREATE TABLE 'db'.'table' (config VARCHAR(256) NOT NULL, task_id INT NOT NULL, subtask_id
INT, scn DECIMAL(20, 0) NOT NULL, scn_low DECIMAL(20, 0), cycle_id INT, table_name
VARCHAR(256), status VARCHAR(256));
```

Oracle:

```
CREATE TABLE "owner"."table" (config VARCHAR(256) NOT NULL, task_id INT NOT NULL,
subtask_id INT, scn NUMBER NOT NULL, scn_low NUMBER, cycle_id INT, table_name
VARCHAR(256), status VARCHAR(256));
```

Vertica

```
CREATE TABLE "owner"."table" (config VARCHAR(256) NOT NULL, task_id INT NOT NULL,
subtask_id INT, scn DECIMAL(20, 0) NOT NULL, scn_low DECIMAL(20, 0), cycle_id INT,
table_name VARCHAR(256), status VARCHAR(256));
```

APPENDIX G

Sample Scripts for Enabling or Disabling SQL Server Change Data Capture

If you use Microsoft SQL Server Enterprise Edition sources and do not run the Server Manager on the source database system, you must enable SQL Server Change Data Capture manually by running an SQL script before you create a configuration. Use the sample scripts in this appendix as a basis for creating a script that enables or disables Change Data Capture.

Copy and customize the sample script for your SQL Server database. Replace the following variables with the values that reflect your environment:

- *database_name* represents the name of an SQL Server database.
- *owner_name* represents the name of a table owner.
- *table_name* represents the name of a table.

Microsoft SQL Server Enterprise Edition

Script for Enabling Change Data Capture

Use the following sample script to enable Microsoft SQL Server Change Data Capture for the database and for each source table:

```
-- Repeat the following database block for each database for which to enable Change Data Capture.
-- BEGIN DATABASE BLOCK
use database_name
EXECUTE sys.sp_cdc_enable_db
-- In each database block, repeat the following table block for each table in the database
-- for which to enable Change Data Capture.
-- BEGIN TABLE BLOCK
execute sys.sp_cdc_enable_table
    @source_schema = N'owner_name'
  , @source_name = N'table_name'
  , @role_name = NULL
-- END TABLE BLOCK
-- END DATABASE BLOCK
EXEC sp_configure 'max text repl size', 2147483647
RECONFIGURE WITH OVERRIDE
```

Script for Disabling Change Data Capture

Use the following sample script to disable Microsoft SQL Server Change Data Capture for the database and for each source table:

```
-- Repeat the following database block for each database for which to disable Change Data Capture.
-- BEGIN DATABASE BLOCK
use database_name
-- In each database block, repeat the following table block for each table in the database
-- for which to disable Change Data Capture.
-- BEGIN TABLE BLOCK
execute sys.sp_cdc_disable_table
    @source_schema = N'owner_name'
  , @source_name = N'table_name'
  , @capture_instance = 'all'
-- END TABLE BLOCK
-- END DATABASE BLOCK
```

Alternatively, to disable Microsoft SQL Server Change Data Capture for all of the tables in a particular database, use the following script:

```
use database_name
EXECUTE sys.sp_cdc_disable_db
```

APPENDIX H

Glossary

Applier

A Data Replication component that reads the committed changes from the generated intermediate files and applies these changes to the target.

Applier task

A Server Manager task that runs the Applier component on the Server Manager instance that you specify.

apply cycle

A cycle of applying change data that starts with parsing the intermediate files by the Applier and ends with the commit of the data to the target. For continuous replication, the Applier processes the data in multiple low-latency apply cycles. For batched replication, the Applier by default processes the data in one apply cycle.

Audit Apply

An apply mode that you can use to audit changes for regulatory or oversight purposes. Data Replication uses an SQL insert operation to add a row in the audit log table on the target for each update, insert, and delete that occurs on the source. For Apache Kafka, Cloudera, Flat File, and Hortonworks targets, Audit Apply is the only available apply mode that you can use to replicate data and metadata to Kafka messages or an output file.

audit log table

For Audit and Merge Apply modes used for target types other than Apache Kafka, Cloudera, Flat File, and Hortonworks, a table on the target system that you can use to audit change activity. The table contains a row for each committed DML operation on the source, and _OLD and _NEW columns that contain the before and after values for each mapped source column, as appropriate for the operation type. Also, each row includes metadata about the DML operation such as the timestamp, SCN or LSN, and transaction ID. By default, the name of the audit log table matches the name of the corresponding source table followed by the _LOG suffix.

bidirectional replication

A type of replication that replicates change data between two databases in both directions simultaneously. For bidirectional replication, Data Replication maintains data integrity by providing loopback avoidance and conflict resolution.

binary log

In MySQL, a series of binary log files that contain events that describe data and DDL changes for a database. The binary log provides a record of the events that occur after a backup is made for recovery purposes. The term *binary log file* refers to an individual log file that has a unique name consisting of a base name, "_binlog,"

and a generated numeric suffix. The term *binary log* refers to the series of binary log files with the same base name plus a corresponding index file that contains the names of the binary log files that have been used.

On the **Extract Range** tab of the Data Replication Console, you specify the path to the directory that contains the binary log files.

calculated columns

In the audit log tables, columns that contain metadata about DML operations on the source including the code for the SQL operation type (I for insert, U for update, and D for delete), timestamp, SCN, transaction ID, commit SCN, and commit time. On the **Runtime Settings** tab > **Calculated Columns** view, you can override the default names for these columns.

cascade replication

A type of replication that replicates change data unidirectionally down a chain of databases. Databases in the middle of the chain act as both a source and target simultaneously. From any database in the chain, Data Replication replicates forward only the changes that originated on the database and skips any changes that were received from a preceding database in the chain.

change data capture

Extracts the change data for the mapped tables and columns from the database transactional log.

checkpoint processing

A mechanism that Data Replication uses to recover from unplanned exceptions, failures, and planned outages that disrupt replication. By separately recording checkpoint information for the Extractor, Applier, and Server Manager components, Data Replication can prevent data loss and ensure data consistency across all of the replication stages.

Command Line Interface

See *Replication Configuration Command Line Interface* on page 462 and *Server Manager Command Line Interface* on page 463.

configuration file

See *replication configuration file* on page 462.

continuous replication

A type of replication that delivers source table changes continuously to the target with very low latency. To implement a continuous replication, you must define a schedule that sets the **Run** option to Continuous. When you run the schedule, the Extractor, Send File, and Applier tasks remain active until you stop them or stop the schedule. On the **Runtime Settings** tab > **General** view, you can specify the latency for continuous replication in the **Continuous Replication Latency** field.

Copy File task

A Server Manager task that copies files from the source Server Manager instance to the target Server Manager instance.

data files

The Data Replication .dat files, which together with the .trn files, comprise the intermediate files. The .dat files contain the transactional data changes. See also *transaction files* on page 464.

data warehouse appliance

An integrated set of servers, storage, and operating systems that are pre-installed and pre-optimized for data warehousing. As database warehouse appliances, Data Replication supports Greenplum, Netezza, Oracle Exadata, Teradata, and Vertica targets.

Edit mode

In the Informatica Data Replication Console, a mode that allows you to edit the replication settings of the configuration file. In this mode, running replication tasks cannot concurrently modify the file. See also *Read mode* on page 461.

External task

A Server Manager task that runs an external program on the Server Manager instance that you specify.

Extractor

A Data Replication component that reads the transaction logs of the source database system and extracts data changes such as inserts, updates, and deletes for the selected source tables, as defined in the configuration file.

Extractor task

A Server Manager task that runs the Extractor component on the Server Manager instance that you specify.

Flat File target

A target type that you can use to write the captured change data to text files instead of to a target database.

global transaction

A transaction that contains the changes that are distributed across multiple Applier threads. These threads can be Applier task or subtask threads.

heterogeneous replication

The replication of change data between different types of databases that might also run on different hardware platforms and operating systems.

Informatica Data Replication Console

The graphical user interface that you use to configure, administer, deploy, monitor, schedule, and manage Data Replication replication jobs.

initial materialization

A process of loading a target with data prior to starting change extraction and replication.

InitialSync

A Data Replication component that materializes target tables from source tables. You can use it to perform an initial load of data prior to starting change data replication for the first time using SQL or Merge Apply mode. You do not need to perform an initial load of audit log tables.

As an alternative to InitialSync, for Oracle sources only, you can use Informatica Fast Clone to perform a high-speed initial load of target tables.

InitialSync task

A Server Manager task that runs the InitialSync component on the Server Manager instance that you specify.

intermediate files

Files that the Extractor generates on disk to store the captured change data. An intermediate file is composed of a data file (.dat) and a transaction file (.trn). The Applier parses intermediate files and applies the committed changes to the target database.

log coordinates

In MySQL, a value in the format *logfile.numericSuffix*:*logfile_position* that identifies a specific position in the binary log files. Data Replication uses log coordinates to specify the Extractor Start Point and Applier Sync Point.

loopback avoidance

When replicating changes between two databases in both directions simultaneously, a mechanism that prevents changes from being returned to the database to which they were originally written.

Merge Apply

An apply mode that you can use for Greenplum, Netezza, Oracle, Teradata, and Vertica targets. It combines features of the SQL Apply and Audit Apply modes. For each source table, a target table and an audit log table must exist on the target. The Applier accumulates all changes in audit log tables and then merges the changes in small batches into the target by using optimized SQL statements and fewer transactions than on the source. Instead of replicating all of the operations, the Applier applies only the final image for each row.

Microsoft SQL Server Backup task

An optional Server Manager task for Microsoft SQL Server sources that produces SQL Server backup logs in native format on the Server Manager instance that you specify.

primary target

When configuring replication from a single source to multiple targets, the target that you connect to and that is specified on the **Target Database** tab in the Data Replication Console. The primary target provides the target schema for all of the targets. See also *secondary target* on page 462.

Queue Adapter

An Informatica Java-based tool with which the Data Replication Applier integrates to stream change data to an Apache Kafka target messaging system. The Queue Adapter contains Extractor, Formatter, and Connector components that consume data from the source, format the data into Avro messages, and connect to and send messages to the target message queue.

Read mode

In the Informatica Data Replication Console, a mode that you select when you are done entering replication settings and are ready to start the replication tasks. In this mode, you cannot edit the file, but replication tasks can add changes to the file. See also *Edit mode* on page 460.

recovery table

A table in the target database that the Applier creates to store internal information to prevent the Applier from replicating previously applied data again when the Applier task restarts. With the recovery tables and

checkpoint processing in place, the Extractor, Server Manager, and Applier can resume processing correctly after different types of failures and planned outages.

The Applier uses recovery tables for all target types except Cloudera, Flat File, Hortonworks, and Kafka.

Replication Configuration Command Line Interface

A Data Replication command line utility that you can use to generate a replication configuration based on an input XML file that contains basic replication settings. Later, you can use the configuration to run replication tasks from the command line.

replication configuration file

An SQLite database file that contains all of the configuration settings for a replication job. These settings include connection information for the source and target databases, table and column mappings, the location of the database log files, and runtime settings. Extractor, Applier, and InitialSync tasks retrieve the relevant settings from the configuration file and store internal processing information in the file.

replication statistics

For replication tasks, statistical data in graphical and numeric format that you can view in the Data Replication Console to analyze and diagnose potential problems and assess performance.

replication tasks

The tasks in a replication job, including the InitialSync, Extractor, Send File, and Applier tasks. These tasks can be started manually or scheduled to run.

routing

For replication jobs that have multiple targets, the process of identifying additional targets for a replication job on the **Routing** tab of the Informatica Data Replication Console and routing data to these targets.

secondary target

When configuring replication from a single source to multiple targets, a target that you define on the **Routing** tab and that is not the primary target, which provides the target schema. However, for secondary targets, you can override the primary target schema name and table names. See also *primary target* on page 461.

Send File task

A Server Manager task that transmits the intermediate files from the source Server Manager instance to the target Server Manager instance.

Server Manager

A Data Replication component that acts as the central gateway for configuration, management, and monitoring of all replication jobs. The Server Manager enables data movement from a source database to a target database over a network. You must use the Server Manager for the following types of replications:

- Replications across heterogeneous platforms in a distributed environment
- Continuous replication
- Replications with multiple targets or sources

You can also use the Server Manager to schedule replication jobs to run at certain time, periodically or continuously.

See also *Server Manager Main server* on page 463 and *Server Manager subserver* on page 463.

Server Manager Command Line Interface

A Server Manager command line utility that you can use to manage Data Replication configurations, schedules, and tasks.

Server Manager Main server

The master Server Manager server that can run on any system in the distributed replication topology. The Main server maintains the replication configurations and schedules and manages the distributed replication tasks. See also *Server Manager subserver* on page 463.

Server Manager subserver

A Server Manager server that provides local task management. It is associated with the Server Manager Main server in the replication environment. A subserver can start an Extractor, Applier, or InitialSync task and manage the Send File task. If you replicate data to multiple targets, you must run a subserver on each system other than the one with the Server Manager Main server. See also *Server Manager Main server* on page 463.

SQL Apply

An apply mode in which the SQL statements that Data Replication executes on the target are equivalent to the SQL change operations that occurred on the source. The Applier applies only the committed transactions in the same order as they occurred on the source using a single transaction.

SQL Script Engine

A Data Replication library that executes user-defined SQL expressions in the target database to calculate values of source virtual columns.

staging table

In Merge Apply mode, see *audit log table* on page 458.

Start Point

A Start Point indicates the point from which the Extractor starts reading changes from the source logs (transaction logs, redo logs, or binary logs, depending on the source type) when you run the Extractor for the first time with a particular configuration. The Start Point is expressed as an SCN for Oracle, an LSN for DB2 and Microsoft SQL Server, and a log coordinates value for MySQL.

subtask threads

The child Applier threads that you use to distribute the changes of the Applier thread and provide more parallelism in Applier processing.

supplemental logging

The logging of additional information in database log files by a relational source DBMS whenever an SQL change occurs on a source table. Data Replication requires this information to replicate some DDL operations, get the before images and after images for processing Updates, and resolve conflicts. The method of enabling supplemental logging varies by relational source DBMS:

- For DB2 for Linux, UNIX, and Windows sources, supplemental logging is enabled by specifying the DB2 DATA CAPTURE CHANGES option for each source table.
- For Microsoft SQL Server sources, supplemental logging is enabled by enabling SQL Server Change Data Capture for each source table.

- For Oracle sources, supplemental logging is enabled by adding a supplemental log group for each source table. The Data Replication Console can generate supplemental log groups when you save a configuration that includes Oracle source tables. Alternatively, you can add the supplemental log groups. You can also add or remove the table columns that are included in the supplemental log groups.

To enable or disable database supplemental logging, use either the Data Replication Console or the Command Line Interface for the Server Manager.

Sync Point

The Sync Point indicates the point in source logs from which the Applier starts applying changes to the target. After initial synchronization, the Sync Point indicates the point up to which InitialSync synchronized the target tables with the source tables. Data Replication stores a Sync Point value for each mapped table in a configuration. A Sync Point is expressed as an SCN for Oracle, an LSN for DB2 and Microsoft SQL Server, and a log coordinates value for MySQL. You can edit the Sync Point to resynchronize a target table or start apply processing from a specific point.

Tcl expression

An expression that contains Tcl commands for advanced processing of replicated data. You can use Tcl expressions to define transformations for the captured change data.

Tcl Script Engine

A Data Replication library that runs user-defined Tcl scripts to calculate values of source virtual columns in the target database.

transactional replication

A type of data replication in which the SQL operations in a database transaction are replicated as a logical unit of work (UOW) from a source to a target that has a similar schema. This type of replication attempts to maintain transactional integrity. If any SQL operations within the begin and commit boundaries of the UOW fail, the entire transaction is not applied to the target.

transaction files

The Data Replication .trn files, which together with .dat files, comprise the intermediate files. The .trn files contain transaction metadata. See also *data files* on page 459.

virtual column

A column that you define in the source table for assigning a Tcl expression.

virtual index

A set of source columns that you can define to uniquely identify rows in the target tables. You must specify virtual indexes for the source tables that have no primary key or unique index definition for accurate replication of Updates and Deletes.

INDEX

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W