Oracle® Database

Using Oracle GoldenGate with Heterogeneous Databases

21c (21.1.0)
F25356-02
October 2021

Oracle Database Using Oracle GoldenGate with Heterogeneous Databases, 21c (21.1.0)

F25356-02

Primary Author: Oracle Corp.

Contents

Preface

Audience vii
Documentation Accessibility vii
Related Information vii
Conventions viii

Part I What is Oracle GoldenGate for Heterogeneous Databases?

Part II Using Oracle GoldenGate for DB2 for z/OS

1 Understanding What's Supported for DB2 for z/OS

1.1 Supported DB2 for z/OS Data Types 1-1
1.2 Non-Supported DB2 for z/OS Data Types 1-1
1.3 Supported Objects and Operations for DB2 z/OS 1-2
1.4 Non-Supported Objects and Operations for DB2 for z/OS 1-3

2 Preparing the DB2 for z/OS Database for Oracle GoldenGate

2.1 Preparing Tables for Processing 2-1
2.1.1 Disabling Triggers and Cascade Constraints 2-1
2.1.2 Assigning Row Identifiers 2-1
2.1.2.1 How Oracle GoldenGate Determines the Kind of Row Identifier to Use 2-2
2.1.2.2 Using KEYCOLS to Specify a Custom Key 2-2
2.1.3 Handling ROWID Columns 2-2
2.2 Configuring a Database Connection 2-3
2.2.1 Database Configuration for DB2 z/OS 2-3
2.2.2 Database User for Oracle GoldenGate Processes 2-3
2.2.3 Setting Initialization Parameters 2-4
2.2.4 Specifying the Path to the Initialization File 2-5
2.2.5 Ensuring ODBC Connection Compatibility 2-5

2.2.6 Specifying the Number of Connection Threads 2-5
2.3 Monitoring Processes 2-6
2.3.1 Interpreting Statistics for Update Operations 2-6
2.4 Supporting Globalization Functions 2-6
2.4.1 Replicating From a Source that Contains Both ASCII and EBCDIC 2-6
2.4.2 Specifying Multi-Byte Characters in Object Names 2-7

3 Preparing the DB2 for z/OS Transaction Logs for Oracle GoldenGate

3.1 Making Transaction Data Available 3-1
3.1.1 Enabling Change Capture 3-1
3.1.2 Enabling Access to Log Records 3-1
3.1.3 Sizing and Retaining the Logs 3-2
3.1.4 Using Archive Logs on Tape 3-3
3.1.5 Controlling Log Flushes 3-3

Part III Using Oracle GoldenGate for MySQL

4 Understanding What's Supported for MySQL

4.1 Character Sets in MySQL 4-1
4.2 Supported MySQL Data Types 4-1
4.2.1 Limitations and Clarifications 4-2
4.3 Non-Supported MySQL Data Types 4-4
4.4 Supported Objects and Operations for MySQL 4-4
4.4.1 Details of Support for Objects and Operations in MySQL DDL 4-5
4.5 Systems Schemas 4-5

5 Preparing and Configuring the System for Oracle GoldenGate

5.1 Database User for Oracle GoldenGate Processes for MySQL 5-1
5.2 Ensuring Data Availability 5-3
5.3 Setting Logging Parameters 5-3
5.4 Database Connection 5-5
5.5 Setting the Session Character Set 5-5
5.6 Preparing Tables for Processing 5-5
5.6.1 Assigning Row Identifiers 5-6
5.6.1.1 How Oracle GoldenGate Determines the Kind of Row Identifier to Use 5-6
5.6.1.2 Tables with a Primary Key Derived from a Unique Index 5-6
5.6.1.3 How to Specify Your Own Key for Oracle GoldenGate to Use 5-7
5.6.2 Limiting Row Changes in Tables That Do Not Have a Key 5-7

5.6.3 Triggers and Cascade Constraints Considerations 5-7
5.7 Changing the Log-Bin Location 5-8
5.8 Configuring Bi-Directional Replication 5-8
5.9 Configuring MySQL for Remote Capture 5-9
5.10 Configuring a Two-way SSL Connection in MySQL Capture and Delivery 5-10
5.11 Capturing using a MySQL Replication Slave 5-11
5.12 Other Oracle GoldenGate Parameters for MySQL 5-11
5.13 Positioning Extract to a Specific Start Point 5-14

6 Using DDL Replication

6.1 Transaction Log Based DDL Configuration Prerequisites and Considerations 6-1
6.2 Plug-in Based DDL Configuration Prerequisites and Considerations 6-2
6.2.1 Installing DDL Replication 6-3
6.2.2 Using the Metadata Server 6-4
6.2.3 Troubleshooting Plug-in Based DDL Replication 6-4
6.2.4 Upgrading from Plugin-based DDL Replication to Transaction Log-based DDL Replication 6-4
6.2.5 Uninstalling Plug-In Based DDL Replication 6-5
6.3 Using DDL Filtering for Replication 6-5

Part IV Using Oracle GoldenGate for SQL Server

7 Understanding What's Supported for SQL Server

7.1 Supported Objects and Operations for SQL Server 7-1
7.2 Non-Supported Objects and Operations for SQL Server 7-1
7.2.1 Requirements for Table Level DDL Changes 7-2
7.3 Supported SQL Server Data Types 7-3
7.4 Non-Supported SQL Server Data Types and Features 7-5
7.5 System Schemas for SQL Server 7-5

8 Preparing the System for Oracle GoldenGate

8.1 Database User for Oracle GoldenGate Processes for SQL Server 8-1
8.1.1 Extract and Replicat Users for SQL Server 8-1
8.1.2 Amazon RDS User Permissions and Requirements 8-2
8.1.3 User that Enables Supplemental Logging and Other Features 8-3
8.2 Configuring a Database Connection 8-4
8.2.1 Configuring an Extract Database Connection 8-4
8.2.2 Configuring a Replicat Database Connection 8-4

8.2.2.1 Using ODBC or Default OLE DB 8-4
8.2.2.2 Using OLE DB with USEREPLICATIONUSER 8-5
8.2.3 Configuring a Database Connection on Linux 8-7
8.2.4 Configuring a Database Connection on Windows 8-7
8.3 Preparing Tables for Processing 8-8
8.3.1 Disabling Triggers and Cascade Constraints on the Target 8-8
8.3.2 Assigning Row Identifiers 8-9
8.3.2.1 How Oracle GoldenGate Determines the Kind of Row Identifier to Use 8-9
8.3.2.2 Using KEYCOLS to Specify a Custom Key 8-9
8.3.3 Improving IDENTITY Replication with Array Processing 8-10
8.4 Globalization Support 8-10

9 Preparing the Database for Oracle GoldenGate — CDC Capture

9.1 Enabling CDC Supplemental Logging 9-1
9.2 Purging CDC Staging Data 9-3
9.3 Enabling Bi-Directional Loop Detection 9-4

10 Requirements Summary for Capture and Delivery of Databases in an Always On Availability Group

10.1 Database Connection 10-1
10.2 Supplemental Logging 10-1
10.3 Operational Requirements and Considerations 10-2

11 CDC Capture Method Operational Considerations

11.1 Tuning SQL Server Change Data Capture 11-1
11.2 Oracle GoldenGate CDC Object Versioning 11-2
11.3 Valid and Invalid Extract Parameters for SQL Server Change Data Capture 11-3
11.4 Details of the Oracle GoldenGate CDC Cleanup Process 11-4
11.5 Changing from Classic Extract to a CDC Extract 11-5

Preface

This guide helps you get started with using Oracle GoldenGate on heterogeneous database systems supported with this release.

Topics:

- Audience
- Documentation Accessibility
- Related Information
- Conventions

Audience

Using Oracle GoldenGate for Heterogeneous Databases is intended for database and system administrators who are responsible for implementing Oracle GoldenGate and managing the databases for an organization.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Information

The Oracle GoldenGate Product Documentation Libraries are found at:

https://docs.us.oracle.com/en/middleware/goldengate/core/21.1/

The Oracle GoldenGate related product documentation libraries are found at:

https://docs.oracle.com/en/middleware/goldengate/index.html

For additional information on Oracle GoldenGate, refer to:

https://www.oracle.com/middleware/technologies/goldengate.html

https://www.oracle.com/database/technologies/high-availability/oracle-database-maa-best-practices.html

For licensing information, refer to Licensing Information in the *Oracle GoldenGate Licensing Information* guide.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, such as "From the File menu, select **Save**." Boldface also is used for terms defined in text or in the glossary.
italic *`italic`*	Italic type indicates placeholder variables for which you supply particular values, such as in the parameter statement: `TABLE` *`table_name`*. Italic type also is used for book titles and emphasis.
`monospace` `MONOSPACE`	Monospace type indicates code components such as user exits and scripts; the names of files and database objects; URL paths; and input and output text that appears on the screen. Uppercase monospace type is generally used to represent the names of Oracle GoldenGate parameters, commands, and user-configurable functions, as well as SQL commands and keywords.
UPPERCASE	Uppercase in the regular text font indicates the name of a utility unless the name is intended to be a specific case.
`{}`	Braces within syntax enclose a set of options that are separated by pipe symbols, one of which must be selected, for example: `{`*`option1`* `|` *`option2`* `|` *`option3`*`}`.
`[]`	Brackets within syntax indicate an optional element. For example in this syntax, the SAVE clause is optional: `CLEANUP REPLICAT` *`group_name`* `[, SAVE` *`count`*`]`. Multiple options within an optional element are separated by a pipe symbol, for example: `[`*`option1`* `|` *`option2`*`]`.

Part I

What is Oracle GoldenGate for Heterogeneous Databases?

Oracle GoldenGate is a comprehensive software package for real-time data capture and replication in heterogeneous IT environments.

The product set enables high availability solutions, real-time data integration, transactional change data capture, data replication, transformations, and verification between operational and analytical enterprise systems. Oracle GoldenGate brings extreme performance with simplified configuration and management, support for cloud environments, expanded heterogeneity, and enhanced security.

This book is divided into parts so that you can easily find information that is relevant to your environment.

See *Installing Oracle GoldenGate* for system requirements and installation details for each of these databases.

See the *Using Oracle GoldenGate on Oracle Cloud Marketplace* to learn about provisioning and other configurations in the Oracle GoldenGate on Marketplace environment.

What's New in This Guide

For Oracle GoldenGate 21c (21.1.0), you can use the following supported heterogeneous databases. This guide describes tasks related to these databases only.

- DB2 z/OS
- MySQL
- SQL Server

For a full list of supported databases and variations, such as Amazon RDS and Azure database services, view the certification matrix available at:
https://www.oracle.com/middleware/technologies/fusion-certification.html
Each database that Oracle GoldenGate supports has its own requirements and configuration.

Parallel Replicat is supported by all databases available with Oracle GoldenGate 21c (21.1.0) and higher. This guide provides information about this feature for each database.

Part II
Using Oracle GoldenGate for DB2 for z/OS

With Oracle GoldenGate for DB2 for z/OS, you can perform initial loads and capture transactional data from supported DB2 for z/OS versions and replicate the data to a DB2 for z/OS database or other supported Oracle GoldenGate targets, such as an Oracle Database.

Oracle GoldenGate for DB2 for z/OS is installed and runs remotely on Linux, zLinux, or AIX.

Oracle GoldenGate for DB2 for z/OS supports data filtering, mapping, and transformations unless noted otherwise in this documentation.

Topics:

- Understanding What's Supported for DB2 for z/OS
  This chapter contains information on database and table features supported byOracle GoldenGate for DB2 z/OS.
- Preparing the DB2 for z/OS Database for Oracle GoldenGate
- Preparing the DB2 for z/OS Transaction Logs for Oracle GoldenGate

1

Understanding What's Supported for DB2 for z/OS

This chapter contains information on database and table features supported byOracle GoldenGate for DB2 z/OS.

Topics:

- Supported DB2 for z/OS Data Types
- Non-Supported DB2 for z/OS Data Types
- Supported Objects and Operations for DB2 z/OS
- Non-Supported Objects and Operations for DB2 for z/OS

1.1 Supported DB2 for z/OS Data Types

This section lists the DB2 for z/OS data types that Oracle GoldenGate supports and any limitations of this support.

- Oracle GoldenGate does not perform character set conversion for columns that could contain multi-byte data. This includes `GRAPHIC`, `VARGRAPHIC` and `DBCLOB` data types, as well as `CHAR`, `VARCHAR`, and `CLOB` for tables defined with `ENCODING_SCHEME` of '`M`' (multiple CCSID set or multiple encoding schemes) or '`U`' (Unicode). Such data is only supported if the source and target systems are the same CCSID.
- Oracle GoldenGate supports ASCII, EBCDIC, and Unicode data format. Oracle GoldenGate converts between ASCII and EBCDIC data automatically. Unicode is not converted.
- Oracle GoldenGate supports most DB2 data types except those listed in Non-Supported DB2 for z/OS Data Types.

Limitations of Support

- The support of range and precision for floating-point numbers depends on the host machine. In general, the precision is accurate to 16 significant digits, but you should review the database documentation to determine the expected approximations. Oracle GoldenGate rounds or truncates values that exceed the supported precision.
- Oracle GoldenGate does not support the filtering, column mapping, or manipulation of large objects greater than 4K in size. You can use the full Oracle GoldenGate functionality for objects that are 4K or smaller.
- Oracle GoldenGate supports the default `TIMESTAMP` and the `TIMESTAMP` with `TIMEZONE` to up to 9 digit fractional value, but no further.

1.2 Non-Supported DB2 for z/OS Data Types

This section lists DB2 for z/OS data types that Oracle GoldenGate does not support. Data that is not supported may affect the integrity of the target data in relation to the source data.

- XML
- User-defined types
- Negative dates

1.3 Supported Objects and Operations for DB2 z/OS

This section lists the database objects and types of operations that Oracle GoldenGate supports.

- Parallel Replicat is supported with Oracle GoldenGate for DB2 z/OS.
- Extraction and replication of DML operations on DB2 for z/OS tables that contain rows of up to 512KB in length. This size exceeds the maximum row size of DB2.
- INSERT operations from the IBM LOAD utility are supported for change capture if the utility is run with LOG YES and SHRLEVEL CHANGE, and the source tables that are being loaded have DATA CAPTURE CHANGES enabled (required by Oracle GoldenGate) and are specified in the Oracle GoldenGate Extract configuration. Oracle GoldenGate also supports initial loads with the LOAD utility to instantiate target tables during initial synchronization.
- Oracle GoldenGate supports the maximum number of columns per table, which is supported by the database.
- Oracle GoldenGate supports the maximum column size that is supported by the database.
- Extraction and replication of data that is stored using DB2 data compression (CREATE TABLESPACE COMPRESS YES).
- Capture from temporal history tables is supported.
- TRUNCATE TABLE is supported, but because this command issues row deletes to perform the truncate, they are shown in Oracle GoldenGate statistics as such, and not as a truncate operation. To replicate a TRUNCATE , the Replicat process uses a DELETE operation without a WHERE clause.
- TRUNCATES are always captured from a DB2 for z/OS source, but can be ignored by Replicat if the IGNORETRUNCATES parameter is used in the Replicat parameter file.
- UNICODE columns in EBCDIC tables are supported.
- Supported options with SHOWTRANS

  ```
  SHOWTRANS [transaction_ID]  [COUNT n]
  [DURATION duration unit]
  [FILE file_name] |
  ```

  transaction_ID and count cannot be specified together.

  transaction_ID and duration cannot be specified together.
- Options supported with SKIPTRANS and FORCETRANS:

  ```
  SKIPTRANS transaction_ID
  [FORCE] FORCETRANS transaction_ID  [FORCE]
  ```

1.4 Non-Supported Objects and Operations for DB2 for z/OS

The following objects and operations are not supported by Oracle GoldenGate on DB2 for z/OS:

- Extraction or replication of DDL operations
- Clone tables
- Data manipulation, including compression, that is performed within user-supplied DB2 exit routines, such as:
  - Date and time routines
  - Edit routines (`CREATE TABLE EDITPROC`)
  - Validation routines (`CREATE TABLE VALIDPROC`)
- Replicating with `BATCHSQL` is not fully functional for DB2 for z/OS. Non-insert operations are not supported so any update or delete operations will cause Replicat to drop temporarily out of `BATCHSQL` mode. The transactions will stop and errors will occur.

2

Preparing the DB2 for z/OS Database for Oracle GoldenGate

Learn how to prepare your database and environment to support Oracle GoldenGate.
Topics:

- Preparing Tables for Processing
- Configuring a Database Connection
- Monitoring Processes
- Supporting Globalization Functions

2.1 Preparing Tables for Processing

You must perform the following tasks to prepare your tables for use in an Oracle GoldenGate environment.

- Disabling Triggers and Cascade Constraints
- Assigning Row Identifiers
- Handling ROWID Columns

2.1.1 Disabling Triggers and Cascade Constraints

Disable triggers, cascade delete constraints, and cascade update constraints on the target tables, or alter them to ignore changes made by the Oracle GoldenGate database user. Oracle GoldenGate replicates DML that results from a trigger or cascade constraint. If the same trigger or constraint gets activated on the target table, it becomes redundant because of the replicated version, and the database returns an error. Consider the following example, where the source tables are `emp_src` and `salary_src` and the target tables are `emp_targ` and `salary_targ`.

- A delete is issued for `emp_src`.
- It cascades a delete to `salary_src`.
- Oracle GoldenGate sends both deletes to the target.
- The parent delete arrives first and is applied to `emp_targ`.
- The parent delete cascades a delete to `salary_targ`.
- The cascaded delete from `salary_src` is applied to `salary_targ`.
- The row cannot be located because it was already deleted in step 5.

2.1.2 Assigning Row Identifiers

Oracle GoldenGate requires some form of unique row identifier on the source and target tables to locate the correct target rows for replicated updates and deletes.

- How Oracle GoldenGate Determines the Kind of Row Identifier to Use
- Using KEYCOLS to Specify a Custom Key

2.1.2.1 How Oracle GoldenGate Determines the Kind of Row Identifier to Use

Unless a `KEYCOLS` clause is used in the `TABLE` or `MAP` statement, Oracle GoldenGate selects a row identifier to use in the following order of priority:

1. Primary key
2. First unique key alphanumerically that does not contain a timestamp or non-materialized computed column.
3. If none of the preceding key types exist (even though there might be other types of keys defined on the table) Oracle GoldenGate constructs a pseudo key of all columns that the database allows to be used in a unique key, excluding those that are not supported by Oracle GoldenGate in a key or those that are excluded from the Oracle GoldenGate configuration.

> **Note:**
>
> If there are other, non-usable keys on a table or if there are no keys at all on the table, Oracle GoldenGate logs an appropriate message to the report file. Constructing a key from all of the columns impedes the performance of Oracle GoldenGate on the source system. On the target, this key causes Replicat to use a larger, less efficient `WHERE` clause.

2.1.2.2 Using `KEYCOLS` to Specify a Custom Key

If a table does not have one of the preceding types of row identifiers, or if you prefer those identifiers not to be used, you can define a substitute key if the table has columns that always contain unique values. You define this substitute key by including a `KEYCOLS` clause within the Extract `TABLE` parameter and the Replicat `MAP` parameter. The specified key will override any existing primary or unique key that Oracle GoldenGate finds. For more information, see *Reference for Oracle GoldenGate*.

2.1.3 Handling `ROWID` Columns

Any attempt to insert into a target table that includes a column with a data type of `ROWID GENERATED ALWAYS` (the default) will fail with the following ODBC error:

```
ODBC error: SQLSTATE 428C9 native database error -798. {DB2 FOR OS/390}
{ODBC DRIVER}{DSN08015} DSNT408I SQLCODE = -798, ERROR: YOU CANNOT
INSERT A VALUE INTO A COLUMN THAT IS DEFINED WITH THE OPTION GENERATED
ALWAYS. COLUMN NAME ROWIDCOL.
```

You can do one of the following to prepare tables with `ROWID` columns to be processed by Oracle GoldenGate:

- Ensure that any `ROWID` columns in target tables are defined as `GENERATED BY DEFAULT`.

- If it is not possible to change the table definition, you can work around it with the following procedure.

To Work Around `ROWID GENERATE ALWAYS`:

1. For the source table, create an Extract `TABLE` statement, and use a `COLSEXCEPT` clause in that statement that excludes the `ROWID` column. For example:

   ```
   TABLE tab1, COLSEXCEPT (rowidcol);
   ```

   The `COLSEXCEPT` clause excludes the `ROWID` column from being captured and replicated to the target table.

2. For the target table, ensure that Replicat does not attempt to use the `ROWID` column as the key. This can be done in one of the following ways:
   - Specify a primary key in the target table definition.
   - If a key cannot be created, create a Replicat `MAP` parameter for the table, and use a `KEYCOLS` clause in that statement that contains any unique columns except for the `ROWID` column. Replicat will use those columns as a key. For example:

     ```
     MAP tab1, TARGET tab1, KEYCOLS (num, ckey);
     ```

2.2 Configuring a Database Connection

This section contains instructions for setting up the Extract and Replicat connections to a SQL Server database.

- Database Configuration for DB2 z/OS
- Database User for Oracle GoldenGate Processes
- Setting Initialization Parameters
- Specifying the Path to the Initialization File
- Ensuring ODBC Connection Compatibility
- Specifying the Number of Connection Threads

2.2.1 Database Configuration for DB2 z/OS

No special DB2 z/OS database settings are required for Oracle GoldenGate.

2.2.2 Database User for Oracle GoldenGate Processes

Oracle GoldenGate requires a database user account. Create this account and assign privileges according to the following guidelines.

Assign the DB2 privileges listed in Table 2-1 to the user by which Extract and Replicat will be running. These are in addition to any permissions that DB2 ODBC requires. All Extract privileges apply to initial-load and log-based Extract processes, except where noted. The following authorities can be provided by granting either `SYSCTRL` or `DBADM` plus `SQLADM` authority to the user running the Oracle GoldenGate processes.

Table 2-1 Privileges Needed by Oracle GoldenGate for DB2 z/OS

User privilege	Extract	Replicat
MONITOR2 (does not apply to initial-load Extract)	X	
SELECT ON the following SYSIBM tables: SYSTABLES SYSCOLUMNS SYSTABLEPART SYSKEYS SYSINDEXES SYSCOLAUTH SYSDATABASE SYSFOREIGNKEYS SYSPARMS SYSRELS SYSROUTINES SYSSYNONYMS SYSTABAUTH SYSAUXRELS	X	X
SELECT on source tables[1]	*X*	
INSERT, UPDATE, DELETE on target tables		X
CREATE TABLE[2]		X
EXECUTE on ODBC plan (default is DSNACLI)	X	
Privileges required by SQLEXEC procedures or queries that you will be using.[3]	X	X

1 SELECT on source tables required only if tables contain LOB columns, or for an initial-load Extract, if used.
2 Required if using ADD CHECKPOINTTABLE in GGSCI to use the database checkpoint feature.
3 SQLEXEC enables stored procedures and queries to be executed by an Oracle GoldenGate process.

2.2.3 Setting Initialization Parameters

The following DB2 for z/OS initialization parameters apply to Oracle GoldenGate and must be set correctly before starting Oracle GoldenGate processes.

- MVSDEFAULTSSID: set to the DB2 subsystem.
- LOCATION: set to the DB2 location name as stored in the DB2 Boot Strap Dataset.
- MVSATTACHTYPE: set to RRSAF (Recoverable Resource Manager Services Attachment Facility) or CAF (Call Attachment Facility). IBM recommends using RRSAF.
- MULTICONTEXT: set to 1 if using RRSAF.
- PLANNAME: set to the DB2 plan. The default plan name is DSNACLI.

Do not use the CURRENTAPPENSCH initialization parameter (keyword).

> **Note:**
>
> When using the `CAF` attachment type, you must use the Oracle GoldenGate `DBOPTIONS` parameter with the `NOCATALOGCONNECT` option in the parameter file of any Extract or Replicat process that connects to DB2. This parameter disables the usual attempt by Oracle GoldenGate to obtain a second thread for the DB2 catalog. Otherwise, you will receive error messages, such as: `ODBC operation failed: Couldn't connect to` *`data source`* `for catalog queries.`

2.2.4 Specifying the Path to the Initialization File

Specify the ODBC initialization file by setting the `DSNAOINI` environment variable in the z/OS UNIX profile, as in the following example:

```
export DSNAOINI="/etc/odbc810.ini"
```

2.2.5 Ensuring ODBC Connection Compatibility

To ensure that you configure the DB2 ODBC initialization file correctly, follow the guidelines in the *DB2 UDB for z/OS ODBC Guide and Reference* manual. One important consideration is the coding of the open and close square brackets (the [character and the] character). The square bracket characters are "variant" characters that are encoded differently in different coded character set identifiers (CCSID), but must be of the IBM-1047 CCSID in the ODBC initialization file. DB2 ODBC does not recognize brackets of any other CCSID. Note the following:

- The first (or open) bracket must use the hexadecimal characters `X'AD'` (`0xAD`).
- The second (or close) bracket must use the hexadecimal characters `X'BD'` (`0xBD`).

To set the correct code for square brackets, use any of the following methods.

- Use the `hex` command in OEDIT and change the hex code for each character appropriately.
- Use the `iconv` utility to convert the ODBC initialization file. For example, to convert from CCSID IBM-037 to IBM-1047, use the following command:

  ```
  iconv -f IBM-037 -t IBM-1047 ODBC.ini > ODBC-1047.ini

  mv ODBC-1047.ini ODBC.ini
  ```

- Change your terminal emulator or terminal configuration to use CCSID IBM-1047 when you create or alter the file.

2.2.6 Specifying the Number of Connection Threads

Every Oracle GoldenGate process makes a database connection. Depending on the number of processes that you will be using and the number of other DB2 connections that you expect, you might need to adjust the following DB2 system parameters on the DSNTIPE DB2 Thread Management Panel:

- `MAX USERS` (macro `DSN6SYSP CTHREAD`)
- `MAX TSO CONNECT` (macro `DSN6SYSP IDFORE`)

- `MAX BATCH CONNECT` (macro `DSN6SYSP IDBACK`)

If using RRSAF, allow:

- Two DB2 threads per process for each of the following:
  - Extract
  - Replicat
  - The GGSCI command `DBLOGIN` (logs into the database)
  - `DEFGEN` utility (generates data definitions for column mapping)
- One extra DB2 thread for Extract for IFI calls.
- One extra DB2 thread for each `SQLEXEC` parameter statement that will be issued by each Extract and Replicat process. For more information about `SQLEXEC`, see the *Reference for Oracle GoldenGate*.

If using CAF, there can be only one thread per Oracle GoldenGate process.

2.3 Monitoring Processes

These sections provide information about monitoring Oracle GoldenGate with z/OS system facilities.

- Interpreting Statistics for Update Operations

2.3.1 Interpreting Statistics for Update Operations

The actual number of DML operations that are executed on the DB2 database might not match the number of extracted DML operations that are reported by Oracle GoldenGate. DB2 does not log update statements if they do not physically change a row, so Oracle GoldenGate cannot detect them or include them in statistics.

2.4 Supporting Globalization Functions

Oracle GoldenGate provides globalization support and you should take into consideration when using this support.

- Replicating From a Source that Contains Both ASCII and EBCDIC
- Specifying Multi-Byte Characters in Object Names

2.4.1 Replicating From a Source that Contains Both ASCII and EBCDIC

When replicating to or from a DB2 source system to a target that has a different character set, some consideration must be given to the encoding of the character data on the DB2 source if it contains a mix of ASCII and EBCDIC data. Character set conversion by any given Replicat requires source data to be in a single character set.

The source character set is specified in the trail header. Thus, the Oracle GoldenGate trail can contain either ASCII or EBCDIC data, but not both. Unicode tables are processed without any special configuration and are exempt from the one-character set requirement.

With respect to a source that contains both character encoding types, you have the following options:

- You can use one Extract for all of your tables, and have it write the character data to the trail as either ASCII or as EBCDIC.
- You can use different Extracts: one Extract to write the ASCII character data to a trail, and another Extract to write the EBCDIC character data to a different trail. You then associate each trail with its own data pump process and Replicat process, so that the two data streams are processed separately.

To output the correct character set in either of those scenarios, use the `TRAILCHARSETASCII` and `TRAILCHARSETEBCDIC` parameters. The default is `TRAILCHARSETEBCDIC`. Without these parameters, ASCII and EBCDIC data are written to the trail as-is. When using these parameters, note the following:

- If used on a single-byte DB2 subsystem, these parameters cause Extract to convert all of the character data to either the ASCII or EBCDIC single-byte CCSID of the subsystem to which Extract is connected, depending on which parameter is used (except for Unicode, which is processed as-is).
- If used on a multi-byte DB2 subsystem, these parameters cause Extract to capture only ASCII or EBCDIC tables (and Unicode). Character data is written in either the ASCII or EBCDIC mixed CCSID (depending on the parameter used) of the DB2 z/OS subsystem to which Extract is connected.

2.4.2 Specifying Multi-Byte Characters in Object Names

If the name of a schema, table, column, or stored procedure in a parameter file contains a multi-byte character, the name must be double-quoted. For more information about specifying object names, see *Administering Oracle GoldenGate*.

3

Preparing the DB2 for z/OS Transaction Logs for Oracle GoldenGate

Learn how to configure the DB2 transaction logging to support data capture by Oracle GoldenGate Extract.

Topics:

- Making Transaction Data Available

3.1 Making Transaction Data Available

Oracle GoldenGate can extract DB2 transaction data from the active and archived logs. Follow these guidelines to configure the logs so that Extract can capture data.

- Enabling Change Capture
- Enabling Access to Log Records
- Sizing and Retaining the Logs
- Using Archive Logs on Tape
- Controlling Log Flushes

3.1.1 Enabling Change Capture

Follow these steps to configure DB2 to log data changes in the expanded format that is supplied by the `DATA CAPTURE CHANGES` feature of the `CREATE TABLE` and `ALTER TABLE` commands. This format provides Oracle GoldenGate with the entire before and after images of rows that are changed with update statements.

1. From the Oracle GoldenGate directory, run GGSCI.
2. Log on to DB2 from GGSCI as a user that has `ALTER TABLE` privileges.

   ```
   DBLOGIN SOURCEDB DSN, USERID user[, PASSWORD password][, encryption_options]
   ```

3. Issue the following command. where *`table`* is the fully qualified name of the table. You can use a wildcard to specify multiple table names but not owner names.

   ```
   ADD TRANDATA table
   ```

   By default, `ADD TRANDATA` issues the following command:

   ```
   ALTER TABLE name DATA CAPTURE CHANGES;
   ```

3.1.2 Enabling Access to Log Records

Activate DB2 Monitor Trace Class 1 ("`TRACE(MONITOR) CLASS(1)`") so that DB2 allows Extract to read the active log. The default destination of `OPX` is sufficient, because Oracle GoldenGate does not use a destination.

To Start the Trace Manually

1. Log on to DB2 as a DB2 user who has the `TRACE` privilege or at least `SYSOPR` authority.
2. Issue the following command:

```
start trace(monitor) class(1) scope(group)
```

To Start the Trace Automatically When DB2 is Started

Do either of the following:

- Set `MONITOR TRACE` to "`YES`" on the `DSNTIPN` installation tracing panel.
- Set '`DSN6SYSP MON=YES` ' in the `DSNTIJUZ` installation job, as described in the *DB2 UDB Installation Guide*.

> **Note:**
>
> The primary authorization ID, or one of the secondary authorization IDs, of the ODBC plan executor also must have the `MONITOR2` privilege.

3.1.3 Sizing and Retaining the Logs

When tables are defined with `DATA CAPTURE CHANGES`, more data is logged than when they are defined with `DATA CAPTURE NONE` . If any of the following is true, you might need to increase the number and size of the active and archived logs.

- Your applications generate large amounts of DB2 data.
- Your applications have infrequent commits.
- You expect to stop Extract for long periods of time.
- Your network is unreliable or slow.

To control log retention, use the `DSN6LOGP MAXARCH` system parameter in the `DSNTIJUZ` installation job.

Retain enough log data so that Extract can start again from its checkpoints after you stop it or after an unplanned outage. Extract must have access to the log that contains the start of the oldest uncommitted unit of work, and all logs thereafter.

If data that Extract needs during processing was not retained, either in online or archived logs, one of the following corrective actions might be required:

- Alter Extract to capture from a later point in time for which log data is available (and accept possible data loss on the target).
- Resynchronize the source and target tables, and then start the Oracle GoldenGate environment over again.

> **Note:**
>
> The IBM documentation makes recommendations for improving the performance of log reads. In particular, you can use large log output buffers, large active logs, and make archives to disk.

3.1.4 Using Archive Logs on Tape

Oracle GoldenGate can read DB2 archive logs on tape, but it will degrade performance. For example, DB2 reserves taped archives for a single recovery task. Therefore, Extract would not be able to read an archive tape that is being used to recover a table until the recovery is finished. You could use DFHSM or an equivalent tools to move the archive logs in a seamless manner between online DASD storage and tape, but Extract will have to wait until the transfer is finished. Delays in Extract processing increase the latency between source and target data.

3.1.5 Controlling Log Flushes

When reading the transaction log, Extract does not process a transaction until it captures the commit record. If the commit record is on a data block that is not full, it cannot be captured until more log activity is generated to complete the block. The API that is used by Extract to read the logs only retrieves full physical data blocks.

A delay in receiving blocks that contain commits can cause latency between the source and target data. If the applications are not generating enough log records to fill a block, Extract generates its own log records by issuing `SAVEPOINT` and `COMMIT` statements, until the block fills up one way or the other and is released.

In a data sharing group, each API call causes DB2 to flush the data blocks of all active members, eliminating the need for Extract to perform flushes.

To prevent Extract from performing flushes, use the Extract parameter `TRANLOGOPTIONS` with the `NOFLUSH` option.

Part III
Using Oracle GoldenGate for MySQL

With Oracle GoldenGate for MySQL, you can perform initial loads and capture transactional data and table changes from supported MySQL versions and replicate the data and table changes to a MySQL database or replicate the data to other supported Oracle GoldenGate targets, such as an Oracle Database.

Oracle GoldenGate for MySQL supports data filtering, mapping, and transformations unless noted otherwise in this documentation.

This part describes tasks for configuring and running Oracle GoldenGate for MySQL and supported variants, such as MariaDB, Amazon RDS for MySQL, and Amazon Aurora MySQL.

Topics:

- Understanding What's Supported for MySQL
  This chapter contains information on database and table features supported by Oracle GoldenGate.
- Preparing and Configuring the System for Oracle GoldenGate
- Using DDL Replication

4
Understanding What's Supported for MySQL

This chapter contains information on database and table features supported by Oracle GoldenGate.

Topics:

- Character Sets in MySQL
- Supported MySQL Data Types
- Non-Supported MySQL Data Types
- Supported Objects and Operations for MySQL
- Systems Schemas

4.1 Character Sets in MySQL

MySQL provides a facility that allows users to specify different character sets at different levels.

Level	Example
Database	`create database test charset utf8;`
Table	`create table test( id int, name char(100)) charset utf8;`
Column	`create table test ( id int, name1 char(100) charset gbk, name2 char(100) charset utf8));`

Limitations of Support

- Binary collations are not supported for multi-byte character sets. For example, do not set the collation_server variable equal to `utf8mb4_bin` when the character set is `utf8mb4`.
- The following character sets are not supported:

  ```
  armscii8
  keybcs2
  utf16le
  geostd8
  ```

4.2 Supported MySQL Data Types

MySQL supports the following data types:

- `BLOB`
- `BIG INT`

- BINARY
- BIT(M)
- CHAR
- DATE
- DATETIME
- DECIMAL
- DOUBLE
- ENUM
- FLOAT
- INT
- JSON
- LONGBLOB
- LONGTEXT
- MEDIUMBLOB
- MEDIUM INT
- MEDIUMTEXT
- SMALL INT
- TEXT
- TIME
- TIMESTAMP
- TINYBLOB
- TINYINT
- TINYTEXT
- VARBINARY
- VARCHAR
- YEAR
- Limitations and Clarifications

4.2.1 Limitations and Clarifications

When running Oracle GoldenGate for MySQL, be aware of the following:

- Functional indexes are not supported for Capture or Delivery.
- Oracle GoldenGate does not support BLOB or TEXT types when used as a primary key.
- Oracle GoldenGate supports UTF8 and UCS2 character sets. UTF8 data is converted to UTF16 by Oracle GoldenGate before writing it to the trail.
- UTF32 is not supported by Oracle GoldenGate.
- Oracle GoldenGate supports a TIME type range from 00:00:00 to 23:59:59.

- Oracle GoldenGate supports timestamp data from `0001/01/03:00:00:00` to `9999/12/31:23:59:59`. If a timestamp is converted from GMT to local time, these limits also apply to the resulting timestamp. Depending on the time zone, conversion may add or subtract hours, which can cause the timestamp to exceed the lower or upper supported limit.
- When the time zone of the Oracle GoldenGate installation server does not match the time zone of the source database server, then the TIMESTAMP data sent to the target database will differ from the source database. For Oracle GoldenGate Microservices installations, regardless of the time zones being the same, Extract will resolve the time zone to UTC. Determine the source database time zone by running the following query:

  ```
  select @@system_time_zone;
  ```

  This will return a time zone value, such as PDT.

  For Classic Architecture, create a session variable for Oracle GoldenGate, called `TZ`, and set it equal to the time zone value of the database.

  For MA, create a variable in the deployment that contains the source Extract, called `TZ` and set it to the value of the source database time zone. After this, stop any running Oracle GoldenGate processes and restart the Administration Service, and then start the Extracts and Replicats.
- Oracle GoldenGate does not support negative dates.
- The support of range and precision for floating-point numbers depends on the host machine. In general, the precision is accurate to 16 significant digits, but you should review the database documentation to determine the expected approximations. Oracle GoldenGate rounds or truncates values that exceed the supported precision.
- When you use `ENUM` type in non-strict `sql_mode`, the non-strict `sql_mode` does not prevent you from entering an invalid `ENUM` value and an error will be returned. To avoid this situation, do one of the following:
  - Use `sql_mode` as `STRICT` and restart Extract. This prevents users from entering invalid values for any of the data types. An IE user can only enter valid values for those data types.
  - Continue using non-strict `sql_mode`, but do not use `ENUM` data types.
  - Continue using non-strict `sql_mode` and use `ENUM` data types with valid values in the database. If you specify invalid values, the database will silently accept them and Extract will abend.
- To preserve transaction boundaries for a MySQL target, create or alter the target tables to the InnoDB transactional database engine instead of the MyISAM engine. MyISAM will cause Replicat records to be applied as they are received, which does not guarantee transaction integrity even with auto-commit turned off. You cannot roll back a transaction with MyISAM.
- Extraction and replication from and to views is not supported.
- Transactions applied by the slave are logged into the relay logs and not into the slave's `binlog`. If you want a slave to write transactions the `binlog` that it receives from the master , you need to start the replication slave with the log slave-updates option as 1 in `my.cnf`. This is in addition to the other binary logging parameters. After the master's transactions are in the slave's `binlog`, you can then setup a regular capture on the slave to capture and process the slave's `binlog`.

4.3 Non-Supported MySQL Data Types

Oracle GoldenGate for MySQL does not support the following data types:

`XML`, `SET`, all spatial types (Geometry and so on).

> **Note:**
>
> Extract abends if it is configured to capture from tables that contain any of the unsupported data types, so ensure that Extract is not configured to capture from tables containing columns of unsupported data types.

4.4 Supported Objects and Operations for MySQL

Oracle GoldenGate for MySQL supports the following objects and operations:

- Oracle GoldenGate supports the following DML operations on source and target database transactional tables:
  - Insert operation
  - Update operation (compressed included)
  - Delete operation (compressed included)
  - Truncate operation
- Oracle GoldenGate supports the extraction and replication of DDL (data definition language) operations.
- Oracle GoldenGate supports transactional tables up to the full row size and maximum number of columns that are supported by MySQL and the database storage engine that is being used. InnoDB supports up to 1017 columns.
- Generated columns are supported and captured.
- Oracle GoldenGate supports the `AUTO_INCREMENT` column attribute. The increment value is captured from the binary log by Extract and applied to the target table in a Replicat insert operation.
- Oracle GoldenGate can operate concurrently with MySQL native replication.
- Oracle GoldenGate supports the `DYNSQL` feature for MySQL.

> **Note:**
>
> XA transactions are not supported for capture and any XA transactions logged in `binlog` cause Extract to abend. So, you must not use XA transactions against a database that Extract is configured to capture. If XA transactions are being used for databases that are not configured for Oracle GoldenGate capture, then exclude those databases from logging into MySQL binary logs by using the parameter `binlog-ignore-db` in the MySQL server configuration file.

Limitations on Automatic Heartbeat Table support are as follows:

- Ensure that the database in which the heartbeat table is to be created already exists to avoid errors when adding the heartbeat table.
- In the heartbeat history lag view, the information in fields like `heartbeat_received_ts`, `incoming_heartbeat_age`, and `outgoing_heartbeat_age` are shown with respect to the system time. You should ensure that the operating system time is setup with the correct and current time zone information.

- Details of Support for Objects and Operations in MySQL DDL

4.4.1 Details of Support for Objects and Operations in MySQL DDL

Here's a list of the MySQL objects and operation types that Oracle GoldenGate supports for the capture and replication of DDL operations.

- DDL replication for MySQL is only supported between MySQL databases as sources and targets.
- Basic extraction and replication of DDL operations are supported for MySQL 5.7.10 and higher.
- For MySQL 5.7.10, only local DDL capture is supported.
- For MySQL 8.0, local and remote DDL capture is supported.
- Only the `CREATE TABLE`, `ALTER TABLE`, and `DROP TABLE` operations are supported.
- `TRUNCATE` operations are supported as DML through the `GETTRUNCATES` Extract and Replicat parameter and do not require configuring Oracle GoldenGate for MySQL DDL support.
- DDL replication is not supported in a Oracle GoldenGate bi-directional configuration.

4.5 Systems Schemas

The following schemas or objects are not automatically replicated by Oracle GoldenGate unless they are explicitly specified without a wildcard.

- `'information_schema'`
- `'performance_schema'`
- `'mysql'`

5

Preparing and Configuring the System for Oracle GoldenGate

Learn about how to prepare the system for running Oracle GoldenGate and how to configure it with your MySQL database.

Topics:

- Database User for Oracle GoldenGate Processes for MySQL
- Ensuring Data Availability
- Setting Logging Parameters
- Database Connection
- Setting the Session Character Set
- Preparing Tables for Processing
- Changing the Log-Bin Location
- Configuring Bi-Directional Replication
- Configuring MySQL for Remote Capture
- Configuring a Two-way SSL Connection in MySQL Capture and Delivery
- Capturing using a MySQL Replication Slave
- Other Oracle GoldenGate Parameters for MySQL
- Positioning Extract to a Specific Start Point

5.1 Database User for Oracle GoldenGate Processes for MySQL

Requirements for the database user for Oracle GoldenGate processes are as follows:

- Create a database user that is dedicated to Oracle GoldenGate. It can be the same user for all the Oracle GoldenGate processes that must connect to a database.
- To support DDL replication, the MySQL user must have privileges to install the database plug-ins. The required permissions for the plug-in is only required with MySQL 5.7. The `INSERT` privilege is required on the `mysql.plugin` system table.
- To preserve the security of your data, and to monitor Oracle GoldenGate processing accurately, do not permit other users, applications, or processes to log on as, or operate as, the Oracle GoldenGate database user.
- Keep a record of the database users. They must be specified in the Oracle GoldenGate parameter files with the `USERID` parameter.
- The Oracle GoldenGate user requires read access to the `INFORMATION_SCHEMA` database.
- The Oracle GoldenGate user requires the following user privileges.

Privilege	Source Extract	Target Replicat	Purpose
SELECT	X	X	Connect to the database and select object definitions
REPLICATION SLAVE	NA	NA	Connect and receive updates from the replication master's binary log.
`REPLICATION CLIENT`	X	NA	Allows to show master, slave, and binary log information.

`CREATE`
`CREATE VIEW`
`EVENT`
`INSERT`
`UPDATE`
`DELETE`
	X	X	Source and target database heartbeat and checkpoint table creation, and data record generation and purging

`DROP`	X	X	Dropping a Replicat checkpoint table or deleting a heartbeat table implementation
`EXECUTE`	X	X	To execute stored procedures.
`INSERT, UPDATE, DELETE` on target tables	NA	X	Apply replicated DML to target objects
DDL privileges on target objects (if using DDL support)	NA	X	Issue replicated DDL on target objects

- To capture binary log events, an Administrator must provide the following privileges to the Extract user:
  - Read and Execute permissions for the directory where the MySQL configuration file (`my.cnf`) is located

    .
  - Read permission for the MySQL configuration file (`my.cnf`).
  - Read and Execute permissions for the directory where the binary logs are located.
  - Read and Execute permission for the `tmp` directory. The `tmp` directory is `/tmp`. The MySQL database connection requires access to the `/tmp/mysql.sock` file for versions prior to MySQL 8.0.

5.2 Ensuring Data Availability

Retain enough binary log data so that if you stop Extract or there is an unplanned outage, Extract can start again from its checkpoints. Extract must have access to the binary log that contains the start of the oldest uncommitted unit of work, and all binary logs thereafter. The recommended retention period is at least 24 hours worth of transaction data, including both active and archived information. You might need to do some testing to determine the best retention time given your data volume and business requirements.

If data that Extract needs during processing was not retained, either in active or backup logs, one of the following corrective actions might be required:

- Alter Extract to capture from a later point in time for which binary log data is available (and accept possible data loss on the target).
- Resynchronize the source and target tables, and then start the Oracle GoldenGate environment over again.

To determine where the Extract checkpoints are, use the `INFO EXTRACT` command. For more information, see INFO EXTRACT in *Command Line Interface Reference for Oracle GoldenGate*.

5.3 Setting Logging Parameters

To capture from the MySQL transaction logs, the Oracle GoldenGate Extract process must be able to find the index file. index file in turn contains the paths of all binary log files.

> **Note:**
>
> Extract expects that all of the table columns are in the binary log. As a result, only `binlog_row_image` set as `full` is supported and this is the default. Other values of `binlog_row_image` are not supported.

In MySQL 5.7, the `server_id` option must be specified along with `log-bin`, otherwise the server will not start. For MySQL 8.0, the `server_id` is enabled by default.

Extract checks the following parameter settings to get this index file path:

1.

2. Extract TRANLOGOPTIONS parameter with the ALTLOGDEST option. If this parameter specifies a location for the log index file, Extract accepts this location over any default that is specified in the MySQL Server configuration file. When ALTLOGDEST is used, the binary log index file must also be stored in the specified directory. This parameter should be used if the MySQL configuration file does not specify the full index file path name, specifies an incorrect location, or if there are multiple installations of MySQL on the same machine. From Oracle GoldenGate 21c onward, ALTLOGDEST parameter is optional. When ALTLOGDEST is not specified, the binary log index and binary log filepaths will be fetched from the database directly.Please note: The paths thus fetched are also subject to same accessibilitychecks as in the existing process.

   To specify the index file path with TRANLOGOPTIONS with ALTLOGDEST, use the following command format on Windows:

   ```
   TRANLOGOPTIONS ALTLOGDEST "C:\Program Files\MySQL\logs\binlog.index"
   ```

   On Linux, use this format:

   ```
   TRANLOGOPTIONS ALTLOGDEST "/mnt/rdbms/mysql/data/logs/binlog.index"
   ```

   To capture from a remote server or in case of remote capture, you only need to specify the REMOTE option instead of the index file path on the remote server. For remote capture on both Windows and Linux, specify the following in the Extract parameter file:

   ```
   TRANLOGOPTIONS ALTLOGDEST REMOTE
   ```

3. The MySQL Server configuration file: The configuration file stores default startup options for the MySQL server and clients. On Windows, the name of the configuration file is my.ini. On other platforms, it is my.cnf. In the absence of TRANLOGOPTIONS with ALTLOGDEST, Extract gets information about the location of the log files from the configuration file. However, even with ALTLOGDEST, these Extract parameters must be set correctly:

   - binlog-ignore-db=oggddl: This prevents DDL logging history table entries in the binlog and is set in the my.cnf or my.ini file.
   - log-bin: This parameter is used to enable binary logging. This parameter also specifies the location of the binary log index file and is a required parameter for Oracle GoldenGate, even if ALTLOGDEST is used. If log-bin is not specified, binary logging will be disabled and Extract returns an error.
   - log-bin-index: This parameter specifies the location of the binary log index. If it is not used, Extract assumes that the index file is in the same location as the log files. If this parameter is used and specifies a different directory from the one that contains the binary logs, the binary logs must not be moved once Extract is started.
   - max_binlog_size: This parameter specifies the size, in bytes, of the binary log file.

   > **Note:**
   >
   > The server creates a new binary log file automatically when the size of the current log reaches the max_binlog_size value, unless it must finish recording a transaction before rolling over to a new file.

- `binlog_format`: This parameter sets the format of the logs. It must be set to the value of `ROW`, which directs the database to log DML statements in binary format. From Oracle GoldenGate 19c onward, Extract silently ignores the `binlog` events that are not written in the `ROW` format instead of abending when it detects a `binlog_format` other than `ROW`.

> **Note:**
>
> MySQL binary logging does not allow logging to be enabled or disabled for specific tables. It applies globally to all tables in the database.

To locate the configuration file, Extract checks the `MYSQL_HOME` environment variable: If `MYSQL_HOME` is set, Extract uses the configuration file in the specified directory. If `MYSQL_HOME` is not set, Extract queries the `information_schema.global_variables` table to determine the MySQL installation directory. If a configuration file exists in that directory, Extract uses it.

4. For MariaDB version 10.2 and later, Oracle GoldenGate works in the same way as for MySQL but a new variable needs to be configured in the `my.cnf` or `my.ini` file. The variable that needs to be added is `"binlog-annotate-row-events=OFF".` Restart MariaDB after configuring this variable and then start the Extract process.

5.4 Database Connection

Oracle GoldenGate gets the name of the database it is supposed to connect to from the `SOURCEDB` parameter. To configure the connection for the `SOURCEDB` parameter, use the following format:

```
SOURCEDB dbname@hostname:port, USERID mysqluser, PASSWORD welcome
```

The `dbname` is the name of the MySQL instance, `hostname` is the name or IP address of the MySQL database server, `port` is the port number of the MySQL instance. If using an unqualified host name, that name must be properly configured in the DNS database. Otherwise, use the fully qualified host name, for example `myhost.company.com`.

5.5 Setting the Session Character Set

The `GGSCI`, Extract and Replicat processes use a session character set when connecting to the database. For MySQL, the session character set is taken from the `SESSIONCHARSET` option of `SOURCEDB` and `TARGETDB`. Make certain you specify a session character set in one of these ways when you configure Oracle GoldenGate.

5.6 Preparing Tables for Processing

This section describes how to prepare the tables for processing. Table preparation requires these tasks:

- Assigning Row Identifiers
- Limiting Row Changes in Tables That Do Not Have a Key
- Triggers and Cascade Constraints Considerations

5.6.1 Assigning Row Identifiers

Oracle GoldenGate requires some form of unique row identifier on the source and target tables to locate the correct target rows for replicated updates and deletes.

- How Oracle GoldenGate Determines the Kind of Row Identifier to Use
- Tables with a Primary Key Derived from a Unique Index
- How to Specify Your Own Key for Oracle GoldenGate to Use

5.6.1.1 How Oracle GoldenGate Determines the Kind of Row Identifier to Use

Unless a `KEYCOLS` clause is used in the `TABLE` or `MAP` statement, Oracle GoldenGate selects a row identifier to use in the following order of priority:

1. Primary key
2. First unique key alphanumerically that does not contain a timestamp or non-materialized computed column.
3. If none of the preceding key types exist (even though there might be other types of keys defined on the table) Oracle GoldenGate constructs a pseudo key of all columns that the database allows to be used in a unique key, excluding those that are not supported by Oracle GoldenGate in a key or those that are excluded from the Oracle GoldenGate configuration.

> **Note:**
>
> If there are other, non-usable keys on a table or if there are no keys at all on the table, Oracle GoldenGate logs an appropriate message to the report file. Constructing a key from all of the columns impedes the performance of Oracle GoldenGate on the source system. On the target, this key causes Replicat to use a larger, less efficient `WHERE` clause.

5.6.1.2 Tables with a Primary Key Derived from a Unique Index

In the absence of a primary key on a table, MySQL will promote a unique index to primary key if the indexed column is `NOT NULL`. If there are more than one of these not-null indexes, the first one that was created becomes the primary key. To avoid Replicat errors, create these indexes in the same order on the source and target tables.

For example, assume that source and target tables named `ggvam.emp` each have columns named first, middle, and last, and all are defined as `NOT NULL`. If you create unique indexes in the following order, Oracle GoldenGate will abend on the target because the table definitions do not match.

Source:

```
mysql> create unique index uq1 on ggvam.emp(first);
mysql> create unique index uq2 on ggvam.emp(middle);
mysql> create unique index uq3 on ggvam.emp(last);
```

Target:

```
mysql> create unique index uq1 on ggvam.emp(last);
mysql> create unique index uq2 on ggvam.emp(first);
mysql> create unique index uq3 on ggvam.emp(middle);
```

The result of this sequence is that MySQL promotes the index on the source "first" column to primary key, and it promotes the index on the target "last" column to primary key. Oracle GoldenGate will select the primary keys as identifiers when it builds its metadata record, and the metadata will not match. To avoid this error, decide which column you want to promote to primary key, and create that index first on the source and target.

5.6.1.3 How to Specify Your Own Key for Oracle GoldenGate to Use

If a table does not have one of the preceding types of row identifiers, or if you prefer those identifiers not to be used, you can define a substitute key if the table has columns that always contain unique values. You define this substitute key by including a `KEYCOLS` clause within the Extract `TABLE` parameter and the Replicat `MAP` parameter. The specified key will override any existing primary or unique key that Oracle GoldenGate finds.

5.6.2 Limiting Row Changes in Tables That Do Not Have a Key

If a target table does not have a primary key or a unique key, duplicate rows can exist. In this case, Oracle GoldenGate could update or delete too many target rows, causing the source and target data to go out of synchronization without error messages to alert you. To limit the number of rows that are updated, use the `DBOPTIONS` parameter with the `LIMITROWS` option in the Replicat parameter file. `LIMITROWS` can increase the performance of Oracle GoldenGate on the target system because only one row is processed.

5.6.3 Triggers and Cascade Constraints Considerations

Triggers

Disable triggers on the target tables, or alter them to ignore changes made by the Oracle GoldenGate database user. Oracle GoldenGate replicates DML that results from a trigger. If the same trigger gets activated on the target table, then it becomes redundant because of the replicated version, and the database returns an error.

Cascade Constraints Considerations

Cascading updates and deletes captured by Oracle GoldenGate are not logged in binary log, so they are not captured. This is valid for both MySQL and MariaDB. For example, when you run the delete statement in the parent table with a parent child relationship between tables, the cascading deletes (if there are any) happens for child table, but they are not logged in binary log. Only the delete or update record for the parent table is logged in the binary log and captured by Oracle GoldenGate.

See https://mariadb.com/kb/en/replication-and-foreign-keys/ and https://dev.mysql.com/doc/refman/8.0/en/innodb-and-mysql-replication.html for details.

To properly handle replication of cascading operations, it is recommended to disable cascade deletes and updates on the source and code your application to explicitly delete or update the child records prior to modifying the parent record. Alternatively, you must ensure that the target parent table has the same cascade constraints configured as the source parent table, but this could lead to an out-of-sync condition between source and target, especially in cases of bi-directional replication.

5.7 Changing the Log-Bin Location

Modifying the binary log location by using the `log-bin` variable in the MySQL configuration file might result in two different path entries inside the index file, which could result in errors. To avoid any potential errors, change the log-bin location by doing the following:

1. Stop any new DML operations.
2. Let the extract finish processing all of the existing binary logs. You can verify this by noting when the checkpoint position reaches the offset of the last log.
3. After Extract finishes processing the data, stop the Extract group and, if necessary, back up the binary logs.
4. Stop the MySQL database.
5. Modify the `log-bin` path for the new location.
6. Start the MySQL database.
7. To clean the old log name entries from index file, use `flush master` or `reset master` (based on your MySQL version).
8. Start Extract.

5.8 Configuring Bi-Directional Replication

In a bi-directional configuration, there are Extract and Replicat processes on both the source and target systems to support the replication of transactional changes on each system to the other system. To support this configuration, each Extract must be able to filter the transactions applied by the local Replicat, so that they are not recaptured and sent back to their source in a continuous loop. Additionally, `AUTO_INCREMENT` columns must be set so that there is no conflict between the values on each system.

1. Configure Oracle GoldenGate for high availability or active-active replication according to the instructions in the Propagating DDL in Active-Active (Bidirectional) Configurations in *Using Oracle GoldenGate for Oracle Database*.
2. To filter out Replicat operations in a bi-directional configuration so that the applied operations are not captured and looped back to the source again, take the following steps on each MySQL database:
   - Configure each Replicat process to use a checkpoint table. Replicat writes a checkpoint to this table at the end of each transaction. You can use one global checkpoint table or one per Replicat process See Overview of Replicat in *Understanding Oracle GoldenGate*.
   - Specify the name of the checkpoint table with the `FILTERTABLE` option of the `TRANLOGOPTIONS` parameter in the Extract parameter file. The Extract process will ignore transactions that end with an operation to the specified table, which should only be those of Replicat.

> **Note:**
>
> Although optional for other supported databases as a means of enhancing recovery, the use of a checkpoint table is required for MySQL when using bi-directional replication (and likewise, will enhance recovery).

3. Edit the MySQL server configuration file to set the `auto_increment_increment` and `auto_increment_offset` parameters to avoid discrepancies that could be caused by the bi-directional operations. The following illustrates these parameters, assuming two servers: **ServerA** and **ServerB**.

   ServerA:

   ```
   auto-increment-increment = 2
   auto-increment-offset = 1
   ```

   ServerB:

   ```
   auto-increment-increment = 2
   auto-increment-offset = 2
   ```

5.9 Configuring MySQL for Remote Capture

Oracle GoldenGate remote capture for MySQL, Amazon RDS for MySQL, Amazon Aurora MySQL, Azure Database for MySQL are used to capture transaction log data from a database located remotely to the Oracle GoldenGate installation.

Database Server Configuration

For remote capture to work, configure the MySQL server as follows:

1. Grant access permissions to the Oracle GoldenGate remote capture user.

   Run the following statements against the remote database to create the user and grant the permissions needed for remote capture.

   ```
   mysql > CREATE USER 'username'@'host' IDENTIFIED BY 'Password';
   mysql > GRANT ALL PRIVILEGES ON *.* TO 'username'@'host' WITH GRANT
   OPTION;
   mysql > FLUSH PRIVILEGES;
   ```

2. The `server_id` value of the remote MySQL server should be greater than 0. This value can be verified by issuing the following statement on the MySQL remote server:

   ```
   mysql > show variables like 'server_id';
   ```

   If the `server_id` value is 0, modify the `my.cnf` configuration file to set to a value greater than 0.

Oracle GoldenGate Configuration

Oracle GoldenGate configuration has the following steps:

1. Provide the remote database's connection information in the Extract's parameter file.

   ```
   SOURCEDB remotedb@mysqlserver.company.com:port, USERID username,
   PASSWORD
   password
   ```

2. Add the following parameter to the Extract's parameter file, after the connection information.

   ```
   TRANLOGOPTIONS ALTLOGDEST REMOTE
   ```

Limitations of Oracle GoldenGate Remote Capture for MySQL

Co-existence of Oracle GoldenGate for MySQL remote capture with the MySQL's native replication slave is supported with following conditions and limitations:

- The native replication slave should be assigned a different server_id than the currently running slaves. The slave server_id values can be seen using the following MySQL command on the master server.

  ```
  mysql> show slave hosts;
  ```

  - If the Oracle GoldenGate capture abends with error "`A slave with the same server_uuid or server_id as this slave has connected to the master`", then change the capture's name and restart the capture.
  - If the native replication slave dies with the error "`A slave with the same server_uuid or server_id as this slave has connected to the master`", then change the native replication slave's `server_id` and restart it.
- Remote capture is supported only on the Linux 64-bit platform and not on Windows. But Oracle GoldenGate remote capture on Linux can capture from a MySQL database running on remote Windows machine.

5.10 Configuring a Two-way SSL Connection in MySQL Capture and Delivery

To use the two way SSL in Oracle GoldenGate MySQL capture and delivery, you need to supply the full paths of the certificate authority (`ca.pem`), the client certificate (`client-cert.pem`) and the client key (`client-key.pem`) files to the capture and delivery.

To know more about generating the certificate files, see:

https://dev.mysql.com/doc/refman/5.7/en/creating-ssl-rsa-files-using-mysql.html

You need to provide these paths in the Extract and Replicat parameter files using the `SETENV` parameter.

Following are the `SETENV` environment parameters to set the two-way SSL connection:

- `OGG_MYSQL_OPT_SSL_CA`: Sets the full path of the certification authority.
- `OGG_MYSQL_OPT_SSL_CERT`: Sets the full path of the client certificate.

- `OGG_MYSQL_OPT_SSL_KEY`: Sets the full path of the client key.

In the following example, the MySQL SSL certificate authority, client certificate, and client key paths are set to the Oracle GoldenGate MySQL Extract and Replicat parameter:

```
SETENV (OGG_MYSQL_OPT_SSL_CA='/var/lib/mysql.pem')
SETENV (OGG_MYSQL_OPT_SSL_CERT='/var/lib/mysql/client-cert.pem')
SETENV (OGG_MYSQL_OPT_SSL_KEY='/var/lib/mysql/client-key.pem')
```

5.11 Capturing using a MySQL Replication Slave

You can configure a MySQL replication slave to capture the master's binary log events from the slave.

Typically, the transactions applied by the slave are logged into the relay logs and not into the slave's `binlog`. For the slave to write transactions in its `binlog`, that it receives from the master , you must start the replication slave with the `log-slave-updates` option as `1` in `my.cnf` in conjunction with the other binary logging parameters for Oracle GoldenGate. After the master's transactions are in the slave's `binlog` , you can set up a regular Oracle GoldenGate capture on the slave to capture and process the slave's `binlog`.

5.12 Other Oracle GoldenGate Parameters for MySQL

The following parameters may be of use in MySQL installations, and might be required if non-default settings are used for the MySQL database. Other Oracle GoldenGate parameters will be required in addition to these, depending on your intended business use and configuration.

Parameter	Description
`DBOPTIONS` with `CONNECTIONPORT` *`port_number`*	Required to specify to the VAM the TCP/IP connection port number of the MySQL instance to which an Oracle GoldenGate process must connect if MySQL is not running on the default of 3306. `DBOPTIONS CONNECTIONPORT 3307`
`DBOPTIONS` with `HOST` *`host_id`*	Specifies the DNS name or IP address of the system hosting MySQL to which Replicat must connect.
`DBOPTIONS` with `ALLOWLOBDATATRUNCATE`	Prevents Replicat from abending when replicated LOB data is too large for a target MySQL `CHAR`, `VARCHAR`, `BINARY` or `VARBINARY` column.

Parameter	Description

SOURCEDB with USERID and PASSWORD	Specifies database connection information consisting of the database, user name and password to use by an Oracle GoldenGate process that connects to a MySQL database. If MySQL is not running on the default port of 3306, you must specify a complete connection string that includes the port number: SOURCEDB *dbname@hostname:port*, USERID *user*, PASSWORD *password*.Example:
	`SOURCEDB mydb@mymachine:3307, USERID myuser, PASSWORD mypassword`
	If you are not running the MySQL database on port 3306, you must also specify the connection port of the MySQL database in the DBLOGIN command when issuing commands that affect the database through GGSCI:
	`DBLOGIN SOURCEDB dbname@hostname:port, USERID user, PASSWORD password`
	For example:
	`GGSCI> DBLOGIN SOURCEDB mydb@mymachine:3307, USERID myuser, PASSWORD mypassword`

SQLEXEC	To enable Replicat to bypass the MySQL connection timeout, configure the following command in a SQLEXEC statement in the Replicat parameter file. `SQLEXEC "select CURRENT_TIME();" EVERY n MINUTES` **Where**: *n* is the maximum interval after which you want Replicat to reconnect. The recommended connection timeout 31536000 seconds (365 days).

Parameter	Description
Global variable `sql_mode`	For heartbeattable to work in MySQL 5.7 , MySQL global variable `sql_mode` should not have `NO_ZERO_IN_DATE`, `NO_ZERO_DATE`. In the following example `sql_mode` includes `NO_ZERO_IN_DATE,NO_ZERO_DATE` values: `mysql> show variables like '%sql_mode%';+---------------+-+ | Variable_name | Value | +--------------- +-- + | sql_mode | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_ DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION | +--------------- +-- ---------- + 1 row in set (0.00 sec)`  These values must be removed by issuing the following command:  `mysql> Set global sql_mode='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,ERROR_FOR_DIVISION_BY_ZERO,NO _AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION'; Query OK, 0 rows affected, 1 warning (0.00 sec)`  `mysql> show variables like '%sql_mode%'; +--------------- +---------------------------------- ----------+ | Variable_name | Value | +--------------- +---------------------------------- ----------+ | sql_mode | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,ERROR_FOR_DIV`

Parameter	Description
	ISION_BY_ZERO,NO_AUTO_CREA TE_USER,NO_ENGINE_SUBSTITUTION | +--------------+---------- ----------+ 1 row in set (0.01 sec)

5.13 Positioning Extract to a Specific Start Point

You can position the Extract to a specific start point in the transaction logs using the `ADD/ALTER EXTRACT` commands:

`{ADD | ALTER EXTRACT}` *`group`*`, LOGNUM` *`log_num`*`, LOGPOS` *`log_pos`*

- *`group`* is the name of the Oracle GoldenGate Extract group for which the start position is required.
- `LOGNUM` is the log file number. For example, if the required log file name is `test.000034`, the `LOGNUM` value is 34. Extract will search for this log file.

> **Note:**
>
> In Microservices Architecture, `ADD EXTRACT` will fail if the `LOGNUM` value contains zeroes preceding the value. For example, `ADD EXTRACT ext1, TRANLOG, LOGNUM 000001, LOGPOS 0` will fail. Instead, set `LOGNUM` to 1 for this example to succeed.

- *`LOGPOS`* is an event offset value within the log file that identifies a specific transaction record. Event offset values are stored in the header section of a log record. To position at the beginning of a `binlog` file, set the `LOGPOS` as 0.

In MySQL logs, an event offset value can be unique only within a given binary file. The combination of the position value and a log number will uniquely identify a transaction record. Maximum Log number length is 8 bytes unsigned integer and Maximum Log offset length is 8 bytes unsigned integer. Log number and Log offset are separated by a pipe (‘|’) delimiter. Transactional records available after this position within the specified log will be captured by Extract. In addition, you can position an Extract using a timestamp.

6
Using DDL Replication

Learn how to install, use, configure, and remove DDL replication.
Data Definition Language (DDL) statements (operations) are used to define MySQL database structures or schema. You can use these DDL statements for data replication between MySQL source and target databases. MySQL DDL specifics are found in the MySQL documentation at https://dev.mysql.com/doc/.

Oracle GoldenGate 21c for MySQL has introduced a transaction log based replication solution for MySQL 8.0, which has improved performance and usability when compared to the plug-in based DDL replication approach.

Topics:

- Transaction Log Based DDL Configuration Prerequisites and Considerations
- Plug-in Based DDL Configuration Prerequisites and Considerations
- Using DDL Filtering for Replication

6.1 Transaction Log Based DDL Configuration Prerequisites and Considerations

The prerequisites for configuring transaction log based DDL replication are as follows:

- Logging of full metadata is mandatory after upgrading to MySQL 8.0 onwards with Oracle GoldenGate 21c and higher. To enable full metadata logging:
  1. Set the value of MySQL server variable `binlog_row_metadata` to `FULL`, inside MySQL configuration file, `my.cnf` for Linux and `my.ini` for Windows.
  2. Restart the database server after changing the configuration file for the settings to take effect.
- DDL replication is not supported in Oracle GoldenGate bi-directional configuration as there is no method to filter the DDL operations to prevent DDL looping.
-
- DDL replication for remote capture is supported for MySQL 8.0 onwards. Transaction log based DDL replication works with both remote or local capture. This was a limitation for earlier Oracle GoldenGate releases. For example, Oracle GoldenGate 19c remote capture did not support DDL replication.
- Transaction log based DDL replication can handle DDLs issued within stored procedures, which is a limitation with plugin-based DDL replication.
- By design, the heartbeat table DDLs are ignored by the capture and you should create the heartbeat tables manually at the target.

6.2 Plug-in Based DDL Configuration Prerequisites and Considerations

This is an older approach to performing DDL Replication. The prerequisites for configuring DDL replication are as follows:

- DDL replication is supported for MySQL 5.7.
- DDL replication is not supported in a Oracle GoldenGate bi-directional configuration as there is no method to filter the DDL operations to prevent DDL looping.
- Remote capture for MySQL 5.7 doesn't support DDL replication.
- Oracle GoldenGate DDL replication uses two plug-ins as a shared library, `ddl_rewriter` and `ddl_metadata`, which must be installed on your MySQL server before Oracle GoldenGate replication starts.
- The standalone application, Oracle GoldenGate `metadata_server`, must be running to capture the DDL metadata.
- The `history` table under the new `oggddl` database (`oggddl.history`). This metadata history table is used to store and retrieve the DDL metadata history. The history table records must be ignored from being logged into the binary log so you must specify `binlog-ignore-db=oggddl` in the `my.cnf` file.
- You should not manually drop the `oggddl` database or the `history` table because all DDL statements that run after this event will be lost.
- You should not stop the `metadata_server` during DDL capture as all the DDL statements that run after this event will be lost.
- You should not manually remove the `ddl_rewriter` and the `ddl_metadata` plugins during DDL capture because all DDL statements that run after this event will be lost.
- DDL executed within the stored procedure is *not* supported. For example , a DDL executed as in the following is *not* supported.

```
CREATE PROCEDURE atssrc.generate_data()
BEGIN
DECLARE i INT DEFAULT 0;
WHILE i < 800 DO
SET i = i + 1;
IF (i = 100) then
alter table atssrc.`ddl6` add col2 DATE after id;
ELSEIF (i = 200) then
alter table atssrc.`ddl6` add col3 DATETIME after datetime;
ELSEIF (i = 300) then
alter table atssrc.`ddl6` add `col4` timestamp NULL DEFAULT NULL
after
channel;
ELSEIF (i = 400) then
alter table atssrc.`ddl6` add col5 YEAR after value;
END IF;
END WHILE;
END$$
```

```
DELIMITER ;
call atssrc.generate_data();
```

- By design, the heartbeat table DDLs are ignored by the capture and you should create the heartbeat tables manually at the target.
- Installing DDL Replication
- Using the Metadata Server
- Troubleshooting Plug-in Based DDL Replication
- Upgrading from Plugin-based DDL Replication to Transaction Log-based DDL Replication
- Uninstalling Plug-In Based DDL Replication

6.2.1 Installing DDL Replication

To install DDL replication, you run the installation script that is provided with Oracle GoldenGate as the replication user. This user must have `Create`, `Insert`,`Select`, `Delete`, `Drop`, and `Truncate` database privileges. Additionally, this user must have write permission to copy the Oracle GoldenGate plugin in the MySQL plugin directory. For example, the MySQL plugin are typically in `/usr/lib64/mysql/plugin/`.

The installation script options are `install`, `uninstall`, `start`, `stop`, and `restart`.

The command to install DDL replication uses the install option, user id, password, and port number respectively:

```
bash-3.2$ ./ddl_install.sh install-option user-id password port-number
```

For example:

```
bash-3.2$ ./ddl_install.sh install root welcome 3306
```

The DDL replication installation script completes the following tasks:

1. Ensures that you have a supported MySQL server version installed. DDL replication is supported for MySQL 5.7.10 and greater.
2. Locates the MySQL plugin directory.
3. Ensures that the `ddl_rewriter`, `ddl_metadata` plugins and the `metadata_server` files exist. If these files are not found, then an error message appears and the installation exits.
4. Ensures that the plugins are already installed. If installed, the script exits with a message requesting you to uninstall first and then reinstall.
5. Stops the `metadata_server` if it is running.
6. Deletes the `oggddl.history` table if it exists.
7. Starts the `metadata_server` as a daemon process.
8. Installs the `ddl_rewriter` and `ddl_metadata` plugins.

6.2.2 Using the Metadata Server

You can use the following options with the metadata server:

- You must have the Oracle GoldenGate `metadata_server` running to capture the DDL metadata.
- Run the install script with `start` option to start the metadata server.
- Run the install script with `stop` option to stop the metadata server.
- Run the install script with `restart` option to stop the running metadata server and start again.
- Oracle GoldenGate DDL replication uses two plugins as a shared library, `ddl_rewriter` and `ddl_metadata`, both of which must be installed on your MySQL server before Oracle GoldenGate replication starts.
- The `oggddl.history` metadata history table is used to store and retrieve the DDL metadata history.

There is a single history table and metadata server for each MySQL server. If you want to issue and capture DDLs from multiple instances of an Extract process on the same database server at the same time, there is a possibility of conflict between accessing and populating the metadata history table. Oracle recommends that you do not run and capture DDLs using multiple Extract instances on the same MySQL server.

6.2.3 Troubleshooting Plug-in Based DDL Replication

Plug-in based DDL replication relies on a metadata history table and the metadata plugin and server. To troubleshoot when DDL replication is enabled, the history table contents and the metadata plugin server logs are required.
You can use the `mysqldump` command to generate the history table dump using one of the following examples:

```
mysqldump [options] database [tables]
mysqldump [options] --databases [options] DB1 [DB2 DB3...]
mysqldump [options] --all-databases [options]
```

For example, `bash-3.2$ mysqldump -uroot -pwelcome oggddl history > outfile`

The metadata plugins and server logs are located in the MySQL and Oracle GoldenGate installation directories respectively.

If you find an error in the log files, you need to ensure that the metadata server is running.

6.2.4 Upgrading from Plugin-based DDL Replication to Transaction Log-based DDL Replication

If you are using the plug-in based solution on MySQL 5.7 and plan to upgrade to MySQL 8.0, which uses transaction log based DDL replication, you need to:

1. Uninstall the plug-in components as mentioned in Uninstalling Plug-In Based DDL Replication
2. Upgrade your database.

3. Re-enable DDL replication support based on the steps provided in Transaction Log Based DDL Configuration Prerequisites and Considerations and check the prerequisites and configuration considerations.

6.2.5 Uninstalling Plug-In Based DDL Replication

If you no longer want to capture the DDL events, then you can use the same install script and select the `uninstall` option to disable the DDL setup. Also, any Extract with DDL parameters should be removed or disabled. If you want to capture the DDL again, you can run the install script again. You should take care when multiple instances of the capture process is running on the same instance of your MySQL server. The DDL setup should *not* be disturbed or uninstalled when multiple capture processes are running and when at most one capture is designed to capture the DDL statement.

Use the installation script with the `uninstall` option to uninstall DDL Replication. For example:

```
bash-3.2$ ./ddl_install.sh uninstall root welcome 3306
```

The script performs the following tasks:

1. Uninstalls the `ddl_rewriter` and `ddl_metadata` plugins.
2. Deletes the `oggddl.history` table if exists.
3. Removes the plugins from MySQL plugin directory.
4. Stops the `metadata_server` if it is running.

6.3 Using DDL Filtering for Replication

The following options are supported for MySQL DDL replication:

Option	Description
`DDL INCLUDE OPTYPE CREATE OBJTYPE TABLE;`	Include create table.
`DDL INCLUDE OBJNAME ggvam.*`	Include tables under the `ggvam`database.
`DDL EXCLUDE OBJNAME ggvam.emp*;`	Exclude all the tables under the `ggvam` database and table name starting with the `emp`wildcard.
`DDL INCLUDE INSTR 'XYZ'`	Include DDL that contains this string.
`DDL EXCLUDE INSTR 'WHY'`	Excludes DDL that contains this string.
`DDL INCLUDE MAPPED`	MySQL DDL uses this option and should be used as the default for Oracle GoldenGate MySQL DDL replication. `DDL INCLUDE ALL` and `DDL` are not supported.
`DDL EXCLUDE ALL`	Default option.

For a full list of options, see DDL in *Reference for Oracle GoldenGate*.

Using DDL Statements and Options

- INCLUDE (default) means include all objects that fit the rest of the description. EXCLUDE means to omit items that fit the description. Exclude rules take precedence over include rules.
- OPTYPE specifies the types of operations to be included or excluded. You can use CREATE and ALTER. Multiple OPTYPE can be specified using parentheses. For example, optype (create, alter). The asterisk (*) wildcard can be specified to indicate all operation types, and this is the default.
- OBJTYPE specifies the TABLE operations to include or exclude. The wildcard can be specified to indicate all object types, and this is the default.
- OBJNAME specifies the actual object names to include or exclude. For example, eric.*. Wildcards are specified as in other cases where multiple tables are specified. The default is *.
- String indicates that the rule is true if any of the strings in stringspec are present (or false if excludestring is specified and the stringspec is present). If multiple string entries are made, at least one entry in each stringspec must be present to make the rule evaluate true.

  For example:

  ```
  ddlops string (“a”, “b”), string (“c”) evaluates true if string “a”
  OR “b” is present, AND string “c” is present
  ```

- local is specified if you want the rule to apply only to the current Extract trail (the Extract trail to which the rule applies must precede this ddlops specification).
- The semicolon is required to terminate the parameter entry.

  For example:

  ```
  ddl optype (create, drop), objname (eric.*);
  ddl exclude objname (eric.tab*);
  exttrail a;
  exttrail b;
  ddl optype (create), objname (joe.*), string (“abc”, “xyz”) local;
  ddl optype (alter), objtype (index);
  ```

  In this preceding example, the exttrail a gets creates and drops for all objects that belong to eric, except for objects that start with tab, exttrail a also gets all alter index statements, unless the index name begins with tab (the rule is global even though it’s included in exttrail b). exttrail b gets the same objects as a, and it also gets all creates for objects that belong to joe when the string abcor xyz is present in the DDL text. The ddlops.c module stores all DDL operation parameters and executes related rules.

Additionally, you can use the DDLOPTIONS parameter to configure aspects of DDL processing other than filtering and string substitution. You can use multiple DDLOPTIONS statements and Oracle recommends using one. If you are using multiple DDLOPTIONS statements, then make each of them unique so that one does not override the other. Multiple DDLOPTIONS statements are executed in the order listed in the parameter file.

See DDL and DDLOPTIONS.

Part IV
Using Oracle GoldenGate for SQL Server

With Oracle GoldenGate for SQL Server, you can perform initial loads and capture transactional data from supported SQL Server versions and replicate the data to a SQL Server database or other supported Oracle GoldenGate targets, such as an Oracle Database.

Oracle GoldenGate for SQL Server supports data filtering, mapping, and transformations unless noted otherwise in this documentation.

This part describes tasks for configuring and running Oracle GoldenGate for SQL Server.

Topics:

- Understanding What's Supported for SQL Server
  This chapter contains information on database and table features supported by Oracle GoldenGate for SQL Server.
- Preparing the System for Oracle GoldenGate
- Preparing the Database for Oracle GoldenGate — CDC Capture
  Learn how to enable supplemental logging in the source database tables that are to be used for capture by the Extract for SQL Server and how to purge older CDC staging data.
- Requirements Summary for Capture and Delivery of Databases in an Always On Availability Group
  Oracle GoldenGate for SQL Server supports capture from a primary replica or a read-only, synchronous mode secondary replica of an Always On Availability Group, and delivery to the primary replica.
- CDC Capture Method Operational Considerations
  This section provides information about the SQL Server CDC Capture options, features, and recommended settings.

7

Understanding What's Supported for SQL Server

This chapter contains information on database and table features supported by Oracle GoldenGate for SQL Server.

Topics:

- Supported Objects and Operations for SQL Server
- Non-Supported Objects and Operations for SQL Server
- Supported SQL Server Data Types
- Non-Supported SQL Server Data Types and Features
- System Schemas for SQL Server

7.1 Supported Objects and Operations for SQL Server

The following objects and operations are supported:

- Oracle GoldenGate supports capture of transactional DML from user tables and delivery to user tables and writeable views.
- `TEXT, NTEXT, IMAGE, VARBINARY, VARBINARY (MAX) VARCHAR (MAX)`, and `NVARCHAR (MAX)` columns are supported in their full size.
- Oracle GoldenGate supports the maximum row sizes that are permitted for tables that are enabled for SQL Server Change Data Capture.
- Oracle GoldenGate supports capture from tables enabled with `PAGE` and `ROW` compression. For partitioned tables that use compression, all partitions must be enabled with the same compression type.
- Oracle GoldenGate supports capture for partitioned tables if the table has the same physical layout across all partitions.
- The sum of all column lengths for a table to be captured from must not exceed the length that SQL Server allows for enabling Change Data Capture for the table. If the sum of all column lengths exceeds what is allowed by the SQL Server procedure `sys.sp.cdc_enable_table`, then `ADD TRANDATA` cannot be added for that table. The maximum allowable record length decreases as more columns are present, so there is an inverse relationship between maximum record length and the number of columns in the table.

7.2 Non-Supported Objects and Operations for SQL Server

The following objects and operations are not supported:

- Parallel Replicat is supported with Oracle GoldenGate for SQL Server.

- For source databases, operations that are not supported by SQL Server Change Data Capture, such as `TRUNCATE` statements. Refer to Microsoft SQL Server Documentation for a complete list of the operations that are limited by enabling SQL Server Change Data Capture.
- Oracle GoldenGate for SQL Server does not support the capture or delivery of DDL changes for SQL Server and extra steps are required for Oracle GoldenGate processes on the source and target to handle any table level DDL changes, including table index rebuild operations. See Requirements for Table Level DDL Changes.
- Views are not supported.
- Operations by the `TextCopy` utility and `WRITETEXT` and `UPDATETEXT` statements. These features perform operations that either are not logged by the database or are only partially logged, so they cannot be supported by the Extract process.
- Partitioned tables that have more than one physical layout across partitions.
- Partition switches against a source table. SQL Server Change Data Capture treats partition switches as DDL operations, and the data moved from one partition to another is not logged in the CDC tables, so you must follow the procedures in Requirements for Table Level DDL Changes to manually implement a partition switch when the table is enabled for supplemental logging.
- Due to a limitation with SQL Server's Change Data Capture, column level collations that are different from the database collation, may cause incorrect data to be written to the CDC tables for character data and Extract will capture them as they are written to the CDC tables. It is recommended that you use `NVARCHAR`, `NCHAR` or `NTEXT` data type for columns containing non-ASCII data or use the same collation for table columns as the database. For more information see, About Change Data Capture (SQL Server).
- Due to a limitation with SQL Server's Change Data Capture, `NOOPUPDATES` are not captured by the SQL Server Change Data Capture agent so there are no records for Extract to capture for no-op update operations.
- Requirements for Table Level DDL Changes

7.2.1 Requirements for Table Level DDL Changes

Oracle GoldenGate for SQL Server, including table index rebuild operations, does not support the capture or delivery of DDL changes for SQL Server. Extra steps are required for Oracle GoldenGate processes on the source and target to handle any table-level DDL changes.

Operations considered to be table level DDL changes include, but are not limited to `ALTER TABLE`, `TRUNCATE`, index rebuilds, and partition switches.

To avoid data inconsistencies due to table level DDL changes, the following steps are required.

1. Source: Pause or Stop application data to the table or tables to be modified.
2. Source: Ensure that there are no open transactions against the table to be modified.
3. Source: Ensure that the SQL Server CDC Capture job processes all remaining transactions for the table that is to be modified.

4. Source: Ensure that the Extract processes all the transactions for the table that is to be modified, prior to making any DDL changes.
5. Target: Ensure that the Replicat processes all the transactions for the table that is to be modified, prior to making any DDL changes.
6. Optionally, implementing an Event Marker table can be used to determine when all of the remaining transactions have been processed for the table that is to be modified, and handle the coordination of when to correctly stop the Extract and Replicat.
7. Source: Stop the Extract process.
8. Target: Stop the Replicat process.
9. Source: Disable supplemental logging for the table to be modified by running `DELETE TRANDATA`.
10. Source: Make table DDL changes to the source table.
11. Target: Make table DDL changes to the target table.
12. Source: Re-enable supplemental logging by running `ADD TRANDATA` to the table(s) after the modifications have been performed.
13. Source: Start the Extract.
14. Target: Start the Replicat.
15. Source: Resume application data to the table or tables that were modified.

7.3 Supported SQL Server Data Types

The following data types are supported for capture and delivery, unless specifically noted in the limitations that follow:

- Binary Data Types
  - `(binary, varbinary, varbinary (max))`
  - `(varbinary (max)`with `FILESTREAM)`
- Character Data Types
  - `(char, nchar, nvarchar, nvarchar (max), varchar, varchar (max))`
- Date and Time Data Types
  - `(date, datetime2, datetime, datetimeoffset, smalldatetime, time)`
- Numeric Data Types
  - `(bigint, bit, decimal, float, int, money, numeric, real, smallint, smallmoney, tinyint)`
- LOBs
  - `(image, ntext, text)`
- Other Data Types
  - `(timestamp, uniqueidentifier, hierarchyid, geography, geometry, sql_variant (Delivery only), XML)`
- Oracle GoldenGate for SQL Server can replicate column data that contains `SPARSE` settings..

Limitations:

- Oracle GoldenGate does not support filtering, column mapping, or manipulating large objects larger than 4KB. Full Oracle GoldenGate functionality can be used for objects of up to 4KB.
- Oracle GoldenGate treats XML data as a large object (LOB), as does SQL Server when the XML does not fit into a row. SQL Server extended XML enhancements (such as lax validation, `DATETIME` , union functionality) are not supported.
- A system-assigned `TIMESTAMP` column or a non-materialized computed column cannot be part of a key. A table containing a `TIMESTAMP` column must have a key, which can be a primary key or unique constraint, or a substitute key specified with a `KEYCOLS` clause in the `TABLE` or `MAP` statements. For more information see Assigning Row Identifiers.
- Oracle GoldenGate supports multibyte character data types and multi byte data stored in character columns. Multibyte data is supported only in a like-to-like, SQL Server configuration. Transformation, filtering, and other types of manipulation are not supported for multibyte character data.
- If capture of data for `TEXT`, `NTEXT`, `IMAGE`, `VARCHAR (MAX)`, `NVARCHAR(MAX)` and `VARBINARY (MAX)` columns will exceed the SQL Server default size set for the `max text repl size` option, extend the size. Use `sp_configure` to view the current value of `max text repl size` and adjust the option as needed.
- Oracle GoldenGate supports UDT and UDA data of up to 2 GB in size. All UDTs except SQL_Variant are supported.
- Common Language Runtime (CLR), including SQL Server built-in CLR data types (such as, geometry, geography, and hierarchy ID), are supported. CLR data types are supported only in a like-to-like SQL Server configuration. Transformation, filtering, and other types of manipulation are not supported for CLR data.
- `VARBINARY (MAX)` columns with the `FILESTREAM` attribute are supported up to a size of 4 GB. Extract uses standard `Win32` file functions to read the `FILESTREAM` file.
- The range and precision of floating-point numbers depends on the host machine. In general, precision is accurate to 16 significant digits, but you should review the database documentation to determine the expected approximations. Oracle GoldenGate rounds or truncates values that exceed the supported precision.
- Oracle GoldenGate supports time stamp data from 0001/01/03:00:00:00 to 9999/12/31:23:59:59. If a time stamp is converted from GMT to local time, these limits also apply to the resulting time stamp. Depending on the time zone, conversion may add or subtract hours, which can cause the time stamp to exceed the lower or upper supported limit.

Limitations on Computed Columns:

- Computed columns, either persisted or non-persisted, are not supported by Microsoft's Change Data Capture. Therefore, no data is written to the trail for columns that contain computed columns. To replicate data for non-persisted computed columns, use the `FETCHCOLS` or `FETCHMODCOLS` option of the `TABLE` parameter to fetch the column data from the table. Keep in mind that there can be discrepancies caused by differences in data values between the time that the column was changed in the data base and the time that Extract fetches the data for the transaction record that is being processed.

- Replicat does not apply DML to any computed column, even if the data for that column is in the trail, because the database does not permit DML on that type of column. Data from a source persisted computed column, or from a fetched non- persisted column, can be applied to a target column that is not a computed column.
- In an initial load, all of the data is selected directly from the source tables, not the transaction log. Therefore, in an initial load, data values for all columns, including non-persisted computed columns, is written to the trail or sent to the target, depending on the method that is used. As when applying change data, however, Replicat does not apply initial load data to computed columns, because the database does not permit DML on that type of column.
- Oracle GoldenGate does not permit a non-persisted computed column to be used in a `KEYCOLS` clause in a `TABLE` or `MAP` statement.
- If a unique key includes a non-persisted computed column and Oracle GoldenGate must use the key, the non-persisted computed column is ignored. This may affect data integrity if the remaining columns do not enforce uniqueness.
- If a unique index is defined on any non-persisted computed columns, it is not used.
- If a unique key or index contains a non-persisted computed column and is the only unique identifier in a table, Oracle GoldenGate must use all of the columns as an identifier to find target rows. Because a non-persisted computed column cannot be used in this identifier, Replicat may apply operations containing this identifier to the wrong target rows.

7.4 Non-Supported SQL Server Data Types and Features

- `SQL_Variant` data type is not supported for capture.
- Tables that contain unsupported data types may cause Extract to Abend. As a workaround, you must remove `TRANDATA` from those tables and remove them from the Extract's `TABLE` statement, or use the Extract's `TABLEEXCLUDE` parameter for the table.

7.5 System Schemas for SQL Server

The following schemas or objects are not be automatically replicated by Oracle GoldenGate unless they are explicitly specified without a wildcard.

- `"sys"`
- `"cdc"`
- `"INFORMATION_SCHEMA"`
- `"guest"`

8
Preparing the System for Oracle GoldenGate

This chapter contains steps to take so that the database with which Oracle GoldenGate interacts is correctly configured to support Oracle GoldenGate capture and delivery. Some steps apply only to a source system, some only to a target, and some to both.
Topics:

- Database User for Oracle GoldenGate Processes for SQL Server
- Configuring a Database Connection
- Preparing Tables for Processing
- Globalization Support

8.1 Database User for Oracle GoldenGate Processes for SQL Server

The following database users and privileges are required for Oracle GoldenGate to capture from and apply to a SQL Server database.

- Extract and Replicat Users for SQL Server
- Amazon RDS User Permissions and Requirements
- User that Enables Supplemental Logging and Other Features

8.1.1 Extract and Replicat Users for SQL Server

The Oracle GoldenGate Extract process captures data from SQL Server tables for initial loads, and from the SQL Server change data capture tables for a change data Extract. The Replicat process applies the data to a target SQL Server database. These processes can use either Windows Authentication or SQL Server Authentication to connect to a database.

- To use Windows authentication, the Extract and Replicat processes inherit the login credentials of the Manager process, as identified by the Log On account specified in the Properties of the Manager service. This account must have the privileges listed here.

Oracle GoldenGate Process	Manager privileges if using the Local System account	Manager privileges if using the local or domain account
Extract (source system)	The `BUILTIN\Administrators` account must be at least a member of the source database role `db_owner`.	The account must be at least a member of the `db_owner` fixed database role of the source database.
Replicat (target system)	The `BUILTIN\Administrators` account must be at least a member of the `db_owner` fixed database role of the target database.	The account must be at least a member of the `db_owner` fixed database role of the target database.

-

To use SQL Server authentication, create a dedicated SQL Server login for Extract and Replicat and assign the privileges listed here.

Extract connecting using SQL Server Authentication	Replicat connecting using SQL Server Authentication
The account must at least be a member of the `db_owner` fixed database role of the source database.	The account must at least be a member of the `db_owner` fixed database role of the target database.

If you are using SQL Server authentication, you must specify the user and password with the `USERID` parameter with the `PASSWORD` option in the Extract or Replicat parameter file. Alternately, you can use the Oracle GoldenGate credential store and specify a user alias with the `USERIDALIAS` parameter.

8.1.2 Amazon RDS User Permissions and Requirements

Here are the steps to set up the permissions for Amzon RDS for SQL Server:

1. Using the Amazon RDS for SQL Server master user name, create a new SQL Server login to be used by Oracle GoldenGate processes.

   ```
   USE [master]
   GO
   CREATE LOGIN [ggs] WITH PASSWORD=N'ggs', DEFAULT_DATABASE=<source
   database>
   GO
   ```

2. Grant the following permissions to the login, based on whether to support capture or delivery for Amazon RDS for SQL Server.

   a. Extract user and `ADD/DELETE TRANDATA`, `ADD/DELETE/ALTER HEARTBEATTABLE`, `Create/Drop` the Oracle GoldenGate CDC cleanup job.

      ```
      USE [msdb]
      GO
      CREATE USER [<user>] FOR LOGIN [<login>];
      grant execute on msdb.dbo.rds_cdc_enable_db to [<user>];
      grant execute on msdb.dbo.rds_cdc_disable_db to [<user>];
      grant select on msdb.dbo.sysjobs to [<user>];
      grant select on msdb.dbo.sysjobactivity to [<user>];
      ALTER ROLE [SQLAgentUserRole] ADD MEMBER [<user>];
      ALTER ROLE [SQLAgentOperatorRole] ADD MEMBER [<user>];
      GO
      USE [<source database>]
      GO
      CREATE USER [<user>] FOR LOGIN [<login>];
      ALTER ROLE [db_owner] ADD MEMBER [<user>];
      GO
      ```

   b. Replicat User and `ADD/DELETE HEARTBEATTABLE TARGETONLY`, `ADD/DELETE CHECKPOINTTABLE`:

      ```
      USE [msdb]
      GO
      CREATE USER [<user>] FOR LOGIN [<login>];
      ```

```
grant select on msdb.dbo.sysjobs to [<user>];
grant select on msdb.dbo.sysjobactivity to [<user>];
ALTER ROLE [SQLAgentUserRole] ADD MEMBER [<user>];
GO
USE [<target database name>]
GO
ALTER ROLE [db_owner] ADD MEMBER [<user>];
GO
```

c. Add a new schema in the database to be used by Oracle GoldenGate objects that may get created in the database, depending on the use case. This schema name needs to be referenced in the GLOBALS file using the parameter `GGSCHEMA`, or if installing the Microservices Architecture, you are prompted during the deployment creation for the schema name.

```
USE [source database,target database]
GO
CREATE SCHEMA [OGG schema name];
```

> **Note:**
>
> Amazon locks down `sp_configure` to the default, you so cannott replicate text data greater than 64MB. You need to use parameter groups.

3. To drop the SQL Server CDC cleanup job in Amazon RDS for SQL Server, and replace with the Oracle GoldenGate CDC cleanup job, run the following script:

```
USE [source database]
GO
EXEC sys.sp_cdc_drop_job N'cleanup';
```

4. Run this script to create the Oracle GoldenGate CDC cleanup job. Enclose the instance endpoint and port connection string in double quotes. The script uses the Microsoft `SQLCMD` utility, so you must ensure that `SQLCMD` is installed on the system where Oracle GoldenGate is installed, as shown in the following example:

```
sh ogg_cdc_cleanup_setup.bat createJob login password source database
"sql2016.samplestring.us-west-1.rds.amazonaws.com,1433" OGG schema name
```

8.1.3 User that Enables Supplemental Logging and Other Features

A database user must issue the `ADD TRANDATA` command to enable supplemental logging on the source database in an Oracle GoldenGate configuration. A database login command (`DBLOGIN`) is issued from GGSCI before `ADD TRANDATA` is issued.

- The database user that enables `TRANDATA` must have `sysadmin` rights.

Extract can run with *dbowner* permissions. However, you also need `sysadmin` rights to issue the `ADD/ALTER/ DELETE/INFO HEARTBEATTABLE` commands, or to create the Oracle GoldenGate CDC Cleanup job using the `ogg_cdc_cleanup_setup.bat` batch file.

8.2 Configuring a Database Connection

This section contains instructions for setting up the Extract and Replicat connections to a SQL Server database.

- Configuring an Extract Database Connection
- Configuring a Replicat Database Connection
- Configuring a Database Connection on Linux
- Configuring a Database Connection on Windows

8.2.1 Configuring an Extract Database Connection

Extract connects to a source SQL Server database through an ODBC (Open Database Connectivity) connection. To create this connection, set up a data source name (DSN) through the Data Sources (ODBC) control panel.

See Configuring a Database Connection on Windows and Configuring a Database Connection on Linux for instructions.

Connecting to the Listener in an Always On Environment

Extract can connect to the Listener of an Always On environment and can be configured to route its read-only queries to an available readable, synchronous mode secondary replica. By default, if Extract connects to a Listener, all processing will be done against the primary replica, but if the Extract is configured with the `TRANLOGOPTIONS ALWAYSONREADONLYROUTING` parameter, its read-only queries are routed by the Listener to an available readable secondary replica. See `TRANLOGOPTIONS` and Requirements Summary for Capture and Delivery of Databases in an AlwaysOn Availability Group for more information.

8.2.2 Configuring a Replicat Database Connection

Replicat can connect to the target database to perform DML operations in the following ways:

- Through ODBC.
- Through OLE DB if the SQL Server driver supports it.
- Through OLE DB as the SQL Server replication user. `NOT FOR REPLICATION` must be set on `IDENTITY` columns, foreign key constraints, and triggers.

Before you select a method to use, review the following guidelines and procedures to evaluate the advantages and disadvantages of each.

- Using ODBC or Default OLE DB
- Using OLE DB with USEREPLICATIONUSER

8.2.2.1 Using ODBC or Default OLE DB

If Replicat connects through ODBC or through the default OLE DB connection, the following limitations apply:

- To keep `IDENTITY` columns identical on source and target when using ODBC or default OLE DB, Replicat creates special operations in its transaction to ensure that the seeds are incremented on the target. These steps may reduce delivery performance.
- You must adjust or disable triggers and constraints on the target tables to eliminate the potential for redundant operations.

To use Replicat with either ODBC or OLE DB, follow these steps:

1. To use ODBC exclusively, include the `DBOPTIONS` parameter with the `USEODBC` option in the Replicat parameter file. (To use the default OLE DB connection, no parameter is required.). For SQL Server CDC for Linux for Oracle GoldenGate, the `USEODBC` option is not allowed.
2. Disable triggers and constraints on the target tables. See Disabling Triggers and Cascade Constraints on the Target.
3. To use `IDENTITY` columns in a bidirectional SQL Server configuration, define the `IDENTITY` columns to have an increment value equal to the number of servers in the configuration, with a different seed value for each one. For example, a two-server installation would be as follows:
   - Sys1 sets the seed value at 1 with an increment of 2.
   - Sys2 sets the seed value at 2 with an increment of 2.

   A three-server installation would be as follows:
   - Sys1 sets the seed value at 1 with an increment of 3.
   - Sys2 sets the seed value at 2 with an increment of 3.
   - Sys3 sets the seed value at 3 with an increment of 3.
4. Configure an ODBC data source. See Configuring a Database Connection on Windows.

8.2.2.2 Using OLE DB with USEREPLICATIONUSER

If Replicat connects as the SQL Server replication user through OLE DB with the `USEREPLICATIONUSER` option, and `NOT FOR REPLICATION` is enabled for `IDENTITY`, triggers with foreign key constraints, the following benefits and limitations apply.

- `IDENTITY` seeds are not incremented when Replicat performs an insert. For SQL Server bidirectional configurations, stagger the seed and increment values like the example in Step 3 of the previous section.
- Triggers are disabled for the Replicat user automatically on the target to prevent redundant operations. However triggers fire on the target for other users.
- Foreign key constraints are not enforced on the target for Replicat transactions. `CASCADE` updates and deletes are not performed. These, too, prevent redundant operations.
- `CHECK` constraints are not enforced on the target for Replicat transactions. Even though these constraints are enforced on the source before data is captured, consider whether their absence on the target could cause data integrity issues.

> **Note:**
>
> Normal `IDENTITY`, trigger, and constraint functionality remains in effect for any users other than the Replicat replication user.

To use Replicat with USEREPLICATIONUSER, follow these steps:

> **Note:**
>
> For SQL Server CDC for Linux for Oracle GoldenGate, the USEREPLICATIONUSER option is not allowed.

> **Note:**
>
> For Replicat, connections using a Microsoft ODBC driver, install the Microsoft OLE DB Driver 18 for SQL Server to support the USEREPLICATIONUSER option:
> https://www.microsoft.com/en-us/download/details.aspx?id=56730

1. In SQL Server Management Studio (or other interface) set the NOT FOR REPLICATION flag on the following objects. For active-passive configurations, set it only on the passive database. For active-active configurations, set it on both databases.
   - Foreign key constraints
   - Check constraints
   - IDENTITY columns
   - Triggers (requires textual changes to the definition; see the SQL Server documentation for more information.)
2. Partition IDENTITY values for bidirectional configurations.
3. In the Replicat MAP statements, map the source tables to appropriate targets, and map the child tables that the source tables reference with triggers or foreign-key cascade constraints. Triggered and cascaded child operations are replicated by Oracle GoldenGate, so the referenced tables must be mapped to appropriate targets to preserve data integrity. Include the same parent and child source tables in the Extract TABLE parameters.

   > **Note:**
   >
   > If referenced tables are omitted from the MAP statements, no errors alert you to integrity violations, such as if a row gets inserted into a table that contains a foreign key to a non-replicated table.

4. In the Replicat parameter file, include the DBOPTIONS parameter with the USEREPLICATIONUSER option.
5. Configure an ODBC data source. See Configuring a Database Connection on Windows.

8.2.3 Configuring a Database Connection on Linux

Oracle GoldenGate for SQL Server database configuration provides the same support for Linux and Windows. However, you need the `msodbcsql13*` or `msodbcsql17*` drivers to set up the connections in a Linux environment.

See:

https://docs.microsoft.com/en-us/sql/connect/odbc/linux-mac/installing-the-microsoft-odbc-driver-for-sql-server?view=sql-server-2017

https://docs.microsoft.com/en-us/sql/connect/odbc/download-odbc-driver-for-sql-server?view=sql-server-2017

The following example demonstrates how to create an ODBC data source in a Linux environment:

1. Create a template file for your data source:

   ```
   vi odbc_template_file.ini
   ```

2. Describe the data source in the template file. In the following example, `myserver_ss2017_source` is used as the name with `DBLOGIN` and `SOURCEDB` to connect to the database.:

   ```
   [myserver_SS2017_source]
   Driver = ODBC Driver 17 for SQL Server
   Server = myserver,1433
   Database = source_database
   User = ssuser
   Password = ssuserpassword
   ```

3. Install the data source using the command:

   ```
   odbcinst -i -s -f odbc_template_file.ini
   ```

   For more information, see:

   https://docs.microsoft.com/en-us/sql/connect/odbc/linux-mac/connection-string-keywords-and-data-source-names-dsns?view=sql-server-2017

8.2.4 Configuring a Database Connection on Windows

Follow these instructions to create a SQL Server system data source name (DSN) for a source or a target SQL Server database. A DSN stores information about how to connect to a SQL Server database through ODBC (Open Database Connectivity).

To create a SQL Server DSN

1. To run the ODBC client, select **Control Panel**, select **Administrative Tools**, and then select **Data Sources (ODBC)**.
2. In the ODBC Data Source Administrator dialog box of the ODBC client, select the **System DSN** tab, and then click **Add**.
3. Under Create New Data Source, select the correct SQL Server driver supported for your version of SQL Server, and then click **Finish**. The Create a New Data Source to SQL Server wizard appears. To determine the correct SQL Server driver, see Database Connection.

4. Supply the following, and then click **Next**:
   - **Name**: Can be of your choosing. In a Windows cluster, use one name across all nodes in the cluster.
   - **Description**: (Optional) Type a description of this data source.
   - **Server**: Select the SQL Server server or instance name. Optionally, the listener\instance name of an Always On Availability Group can be listed.
5. For login authentication, do one of the following, and then click **Next**:
   a. Select **With Integrated Windows Authentication** for Oracle GoldenGate to use Windows authentication.
   b. To use database credentials, **With SQL Server authentication using a login ID and password entered by the user**, and supply login information.
6. If the default database is not set to the one that Oracle GoldenGate will connect to, select **Change the default database to**, and then select the database. Set the other settings to use ANSI. Click **Next**.
7. Leave the next page set to the defaults. Click **Finish**.
8. Click **Test Data Source** to test the connection.
9. If the test is successful, close the confirmation box and the Create a New Data Source box.
10. Repeat this procedure for each SQL Server source and target system.

8.3 Preparing Tables for Processing

The table attributes in the following sections must be addressed in your Oracle GoldenGate environment.

- Disabling Triggers and Cascade Constraints on the Target
- Assigning Row Identifiers
- Improving IDENTITY Replication with Array Processing

8.3.1 Disabling Triggers and Cascade Constraints on the Target

In an environment where SQL Server is the target, consider triggers and cascade constraints that may repeat an operation that occurred on the source. For example, if the source has an insert trigger on TableA that inserts a record into TableB, and Oracle GoldenGate is configured to capture and deliver both TableA and TableB, the insert trigger on the target table, TableA, must be disabled. Otherwise, Replicat inserts into TableA, and the trigger fires and insert into TableB. Replicat will then try to insert into TableB, and then terminate abnormally.

When a trigger or cascade constraint repeats an operation that occurred on the source, you do not have to disable the trigger or constraint when the following conditions are both true:

- You use the `DBOPTIONS USEREPLICATIONUSER` parameter in Replicat.
- You use OLE DB connection for Replicat. The use of the OLE DB connection is the default configuration. Note that the trigger, constraint, or `IDENTITY` property must have `NOT FOR REPLICATION` enabled.

In the following scenario, disable the triggers and constraints on the target:

- Uni-directional replication where all tables on the source are replicated.

In the following scenarios, enable the triggers and constraints on the target:

- Uni-directional replication where tables affected by a trigger or cascade operation are not replicated, and the only application that loads these tables is using a trigger or cascade operation.
- Uni-directional or -bi-directional replication where all tables on the source are replicated. In this scenario, set the target table cascade constraints and triggers to enable `NOT FOR REPLICATION`, and use the `DBOPTIONS USEREPLICATIONUSER` parameter in Replicat.

8.3.2 Assigning Row Identifiers

Oracle GoldenGate requires unique row identifiers on the source and target tables to locate the correct target rows for replicated updates and deletes. Source tables can have any kind of key listed in How Oracle GoldenGate Determines the Kind of Row Identifier to Use. If there is no primary key identified on a table that has fixed-length columns, the length of one of the fixed-length columns must be below 3800 bytes.

- How Oracle GoldenGate Determines the Kind of Row Identifier to Use
- Using KEYCOLS to Specify a Custom Key

8.3.2.1 How Oracle GoldenGate Determines the Kind of Row Identifier to Use

Unless a `KEYCOLS` clause is used in the `TABLE` or `MAP` statement, Oracle GoldenGate selects a row identifier to use in the following order of priority:

1. First unique key alphanumerically that does not contain a timestamp or non-materialized computed column.
2. If neither of these key types exist , Oracle GoldenGate constructs a pseudokey of all columns that the database allows to be used in a unique key, excluding those that are not supported by Oracle GoldenGate in a key or those that are excluded from the Oracle GoldenGate configuration. For SQL Server, Oracle GoldenGate requires the row data in target tables that do not have a primary key to be less than 8000 bytes.

> **Note:**
>
> If there are types of keys on a table or if there are no keys at all on a table, Oracle GoldenGate logs a message to the report file. Constructing a key from all of the columns impedes the performance of Oracle GoldenGate on the source system. On the target, this key causes Replicat to use a larger, less efficient `WHERE` clause.

8.3.2.2 Using `KEYCOLS` to Specify a Custom Key

If a table does not have an applicable row identifier, or if you prefer that identifiers are not used, you can define a substitute key, providing that the table has columns that always contain unique values. You define this substitute key by including a `KEYCOLS` clause within the Extract `TABLE` parameter and the Replicat `MAP` parameter. The specified key overrides any existing primary or unique key that Oracle GoldenGate finds.

8.3.3 Improving `IDENTITY` Replication with Array Processing

Because only one table per session can have `IDENTITY_INSERT` set to `ON`, Replicat must continuously toggle `IDENTITY_INSERT` when it applies `IDENTITY` data to multiple tables in a session. To improve the performance of Replicat in this situation, use the `BATCHSQL` parameter. `BATCHSQL` causes Replicat to use array processing instead of applying SQL statements one at a time.

8.4 Globalization Support

Oracle GoldenGate provides globalization support that lets it process data in its native language encoding. The Oracle GoldenGate apply process (Replicat) can convert data from one character set to another when the data is contained in character column types.

9

Preparing the Database for Oracle GoldenGate — CDC Capture

Learn how to enable supplemental logging in the source database tables that are to be used for capture by the Extract for SQL Server and how to purge older CDC staging data.

You can learn more about CDC Capture with this Oracle By Example:

Using the Oracle GoldenGate for SQL Server CDC Capture Replication http://www.oracle.com/webfolder/technetwork/tutorials/obe/fmw/goldengate/12c/sql_cdcrep/sql_cdcrep.html.

Topics:

- Enabling CDC Supplemental Logging
- Purging CDC Staging Data
- Enabling Bi-Directional Loop Detection

9.1 Enabling CDC Supplemental Logging

With the CDC Extract, the method of capturing change data is via SQL Server Change Data Capture tables, so it is imperative that you follow the procedures and requirements below, so that change data is correctly logged, maintained, and captured by Extract.

You will enable supplemental logging with the `ADD TRANDATA` command so that Extract can capture the information that is required to reconstruct transactions.

`ADD TRANDATA` must be issued for all tables that are to be captured by Oracle GoldenGate, and to do so requires that a valid schema be used in order to create the necessary Oracle GoldenGate tables and stored procedures.

Enabling supplemental logging for a CDC Extract does the following:

- Enables SQL Server Change Data Capture at the database level, if it's not already enabled.
  - `EXECUTE sys.sp_cdc_enable_db`
- Creates a Change Data Capture staging table for each base table enabled with supplemental logging by running `EXECUTE sys.sp_cdc_enable_table`, and creates a trigger for each CDC table. The CDC table exists as part of the system tables within the database and has a naming convention like, `cdc.OracleGG_basetableobjectid_CT`.
- Creates a tracking table of naming convention *`schema`*`.OracleGGTranTables`. This table is used to store transaction indicators for the CDC tables, and is populated when the trigger for a CDC table is fired. The table will be owned by the schema listed in the `GLOBALS` file's, `GGSCHEMA` parameter.
- Creates a unique fetch stored procedure for each CDC table, as well as several other stored procedures that are required for Extract to function. These stored procedures will be owned by the schema listed in the `GLOBALS` file's, `GGSCHEMA` parameter.

- Also, as part of enabling CDC for tables, SQL Server creates two jobs per database:

  ```
  cdc.dbname_capture
  cdc.dbname_cleanup
  ```

- The CDC Capture job is the job that reads the SQL Server transaction log and populates the data into the CDC tables, and it is from those CDC tables that the Extract will capture the transactions. So it is of extreme importance that the CDC capture job be running at all times. This too requires that SQL Server Agent be set to run at all times and enabled to run automatically when SQL Server starts.
- Important tuning information of the CDC Capture job can be found in CDC Capture Method Operational Considerations.
- The CDC Cleanup job that is created by Microsoft does not have any dependencies on whether the Oracle GoldenGate Extract has captured data in the CDC tables or not. Therefore, extra steps need to be followed in order to disable or delete the CDC cleanup job immediately after `TRANDATA` is enabled, and to enable Oracle GoldenGate's own CDC cleanup job. See Retaining the CDC Table History Data for more information.

The following steps require a database user who is a member of the SQL Server System Administrators (`sysadmin`) role.

1. In the source Oracle GoldenGate installation, ensure that a `GLOBALS` (all CAPS and no extension) file has been created with the parameter `GGSCHEMA` *`schemaname`*. Ensure that the schema name used has been created (`CREATE SCHEMA` *`schemaname`*) in the source database. This schema will be used by all subsequent Oracle GoldenGate components created in the database, therefore it is recommended to create a unique schema that is solely used by Oracle GoldenGate, such as `'ogg'`. It is recommended not to use the SQL Server schema **cdc** and to create a new schema specific to Oracle GoldenGate.
2. On the source system, run `GGSCI`
3. Issue the following command to log into the database:

   `DBLOGIN SOURCEDB` *`DSN`* `[,{USERID` *`user`*`, PASSWORD` *`password`* `| USERIDALIAS` *`alias`*`}]`
   Where:

   - `SOURCEDB` *`DSN`* is the name of the SQL Server data source.
   - `USERID` *`user`* is the database login and `PASSWORD` *`password`* is the password that is required if the data source connects via SQL Server authentication. Alternatively, `USERIDALIAS` *`alias`* is the alias for the credentials if they are stored in a credentials store. If using `DBLOGIN` with a DSN that is using Integrated Windows authentication, the connection to the database for the GGSCI session will be that of the user running GGSCI. In order to issue `ADD TRANDATA` or `DELETE TRANDATA`, this user must be a member of the SQL Server `sysadmin` server role.
4. In Admin Client, issue the following command for each table that is, or will be, in the Extract configuration. You can use a wildcard to specify multiple table names.

   `ADD TRANDATA` *`owner.table`*

   `ADD TRANDATA` *`owner`*`.*`

Optionally, you can designate the filegroup in which the SQL Server Change Data Capture staging tables will be placed, by using the `FILEGROUP` option with an existing filegroup name.

```
ADD TRANDATA owner.table FILEGROUP cdctables
```

See `ADD TRANDATA`

9.2 Purging CDC Staging Data

When enabling supplemental logging, data that is required by Extract to reconstruct transactions are stored in a series of SQL Server CDC system tables, as well Oracle GoldenGate objects that are used to track operations within a transaction. And as part of enabling supplemental logging, SQL Server will create its own Change Data Capture Cleanup job that runs nightly by default, and purges data older than 72 hours. The SQL Server CDC Cleanup job is unaware that an Extract may still require data from these CDC system tables and can remove that data before the Extract has a chance to capture it.

If data that Extract needs during processing has been deleted from the CDC system tables, then one of the following corrective actions might be required:

- Alter Extract to capture from a later point in time for which CDC data is available (and accept possible data loss on the target).
- Resynchronize the source and target tables, and then start the Oracle GoldenGate environment over again.

To remedy this situation, Oracle GoldenGate for SQL Server includes the `ogg_cdc_cleanup_setup.bat` program that is used to create an Oracle GoldenGate Cleanup job associated stored procedures and tables.

The Extract, upon startup, will expect, by default, that those Oracle GoldenGate Cleanup task objects exist and will stop if they do not. Extract will issue a warning if the SQL Server CDC Cleanup job exists alongside the Oracle GoldenGate Cleanup job.

The default checks by Extract for the Oracle GoldenGate CDC Cleanup task objects can be overwritten by using the `TRANLOGOPTIONS NOMANAGECDCCLEANUP` in the Extract, but this would only be recommended for development and testing purposes.

Use the following steps immediately after enabling supplemental logging and prior to starting the Extract, to create the Oracle GoldenGate CDC Cleanup job and associated objects. You can re-run these steps to re-enable this feature should any of the objects get manually deleted.

To create the Oracle GoldenGate CDC Cleanup job and objects:

The `ogg_cdc_cleanup_setup` file is located in the the home directory for Classic Architecture and in the `Deployment_Home/etc/conf/ogg` directory for Microservices Architecture.

The script uses the Microsoft `sqlcmd` utility, so ensure that `sqlcmd` is installed on the system where Oracle GoldenGate is installed.

This requires an SQL Server authenticated database user who is a member of the SQL Server System Administrators (`sysadmin`) role. Windows authentication is not supported for the `.bat` script.

1. Stop and disable the database's SQL Server `cdc.dbname_cleanup` job from SQL Server Agent. Alternatively, you can drop it from the source database with the following command.

   ```
   EXECUTE sys.sp_cdc_drop_job 'cleanup'
   ```

2. Run the `ogg_cdc_cleanup_setup.bat` file, providing the following variable values.

   For Windows:

   ```
   ogg_cdc_cleanup_setup.bat createJob userid password databasename
   servername\instancename schema
   ```

   For Linux:

   ```
   ./ogg_cdc_cleanup_setup.sh createJob userid password databasename
   servername,port schema
   ```

   In the preceding examples, USER ID and password should be a valid SQL Server login and password for a user, which has `sysadmin` rights. The `databasename`, `servername\instancename`, or `servername`, `port`, are the source database name, server, and instance, or server and TCP/IP port where SQL Server is running. If only the server name is listed, then the default instance will be connected to. The schema is the schema name listed in the GLOBALS file, with the `GGSCHEMA` parameter. This schema should be the same for all Oracle GoldenGate objects, including supplemental logging, checkpoint tables, heartbeat tables, and the Oracle GoldenGate CDC Cleanup job.

   For example:

   ```
   ogg_cdc_cleanup_setup.bat createJob ggsuser ggspword db1
   server1\inst1 ogg
   ```

   Enclose the connection string and port in double quotes, as shown in the following exmaple:

   ```
   sh ogg_cdc_cleanup_setup.bat createJob login password source
   database "sql2016.samplestring.us-west-1.rds.amazonaws.com,1433"
   OGG schema name
   ```

The Oracle GoldenGate CDC Cleanup job when created, is scheduled to run every ten minutes, with a default retention period of seventy two hours. The job will not purge data for an Extract's recovery checkpoint however, regardless of the retention period.

Additional information of the Oracle GoldenGate CDC Cleanup job can be found in CDC Capture Method Operational Considerations.

9.3 Enabling Bi-Directional Loop Detection

Loop detection is a requirement for bi-directional implementations of Oracle GoldenGate, so that an Extract for one source database does not recapture transactions sent by a Replicat from another source database.

With the CDC Extract capture method, by default, any transaction committed by a Replicat into a database where an Extract is configured, will recapture that transaction from the Replicat as long as supplemental logging is enabled for those tables that the Replicat is delivering to.

In order to ignore recapturing transactions that are applied by a Replicat, you must use the `TRANLOGOPTIONS FILTERTABLE` parameter for the CDC Extract. The table used as the filtering table will be the Oracle GoldenGate checkpoint table that you must create for the Replicat.

To create a Filter Table and enable Supplemental Logging:

The steps below require a database user who is a member of the SQL Server System Administrators (`sysadmin`) role.

1. On the source system, run `GGSCI`
2. Issue the following command to log into the database.

   `DBLOGIN SOURCEDB` *`DSN`* `[,{USERID` *`user`*`, PASSWORD` *`password`* `| USERIDALIAS` *`alias`*`}]`

   In the preceding example, the `SOURCEDB DSN` is the name of the SQL Server data source. The `USERID` user is the database login and `PASSWORD` password is the password that is required if the data source connects through SQL Server authentication. Alternatively, `USERIDALIAS` alias is the alias for the credentials if they are stored in a credentials store. If using `DBLOGIN` with a DSN that is using Integrated Windows authentication, the connection to the database for the GGSCI session is that of the user running GGSCI. In order to issue `ADD TRANDATA` or `DELETE TRANDATA`, this user must be a member of the SQL Server `sysadmin` server role.
3. Create the Oracle GoldenGate checkpoint table that is used by the Replicat to deliver data to the source database.

   Example: `ADD CHECKPOINTTABLE ogg.ggchkpt`

   It is recommended that you use the same schema name as used in the `GGSCHEMA` parameter of the `GLOBALS` file.
4. Enable supplemental logging for the newly created checkpoint table.

   Example: `ADD TRANDATA ogg.ggchkpt`
5. Add the Replicat with the checkpoint table information.

   Example: `ADD REPLICAT reptgt1, EXTTRAIL ./dirdat/e2,checkpointtable` *`ogg.ggchkpt`*
6. Configure the Extract with the `IGNOREREPLICATES` (on by default) and `FILTERTABLE` parameters, using the Replicat's checkpoint table for the filtering table.

   `TRANLOGOPTIONS IGNOREREPLICATES`

   `TRANLOGOPTIONS FILTERTABLE` *`ogg.ggchkpt`*

10

Requirements Summary for Capture and Delivery of Databases in an Always On Availability Group

Oracle GoldenGate for SQL Server supports capture from a primary replica or a read-only, synchronous mode secondary replica of an Always On Availability Group, and delivery to the primary replica.

Topics:

- Database Connection
- Supplemental Logging
- Operational Requirements and Considerations

10.1 Database Connection

For both Extract and Replicat, it is recommended to create a System DSN that uses the Always On Availability Group Listener for the connection.

- For the Replicat, connecting to the Listener allows the Replicat to reconnect if the primary replica performs a failover to a new instance, without having to manually edit the DSN settings to point to the new primary.
- For the Extract connecting to the Listener not only allows reconnecting to the primary without editing the DSN to point to the new instance, but also provides the optional ability to run the Extract's data extraction stored procedures, against a read-only secondary.
- For both Extract and Replicat connected to an Always On environment, use the `AUTORESTART` parameter for the Manager, to restart the processes after a failover.
- To route the Extract's data extraction queries to a read-only secondary, ensure that the DSN connection uses the Listener, that you have one or more read-only secondary replicas that are configured to handle read-only routing, and that the Extract runs with the `TRANLOGOPTIONS ALWAYSONREADONLYROUTING` parameter.
  - Ensure that the Application Intent field of the DSN configuration is set to `READWRITE` and not `READONLY`
  - Refer to the following Microsoft documentation on how to configure read-only routing: https://docs.microsoft.com/en-us/sql/database-engine/availability-groups/windows/configure-read-only-routing-for-an-availability-group-sql-server?view=sql-server-2017

10.2 Supplemental Logging

Supplemental logging must be enabled by normal means (`ADD TRANDATA`) using GGSCI connected to the primary replica and not against a secondary replica.

- Create a DSN to the primary replica, or to the Always On Availability Group Listener, to connect using `DBLOGIN` to run `ADD TRANDATA` from GGSCI.
- The login used to enable supplemental logging must have `sysadmin` membership of the primary replica instance.
- When enabling supplemental logging against the primary replica database, the SQL Server Change Data Capture job does not automatically get created on any secondary replicas. Upon failover from a primary to a secondary, you must manually create the SQL Server Change Data Capture job and the Oracle CDC Cleanup job if in use, on the new primary replica.

  ```
  EXECUTE sys.sp_cdc_add_job N'capture
  ```

  – When creating the SQL Server CDC Capture job on the new primary, the default configuration settings are put in place. So if you have previously modified the default values on the former primary replica, you need to run `sys.sp_cdc_change_job` on the new primary and set the values accordingly.

> **Note:**
>
> Consult the Microsoft documentation on how to enable the CDC Capture job for AlwaysOn Secondary Replicas for more information:.

10.3 Operational Requirements and Considerations

- When an instance is no longer the primary instance but has the SQL Server CDC Capture job installed, the job ceases to run after some time and does not attempt to restart. Upon failover back to that instance, the job does not automatically start, so it must be manually started.
- If secondary replica databases are not in sync with the primary replica database, the CDC capture job will not advance in the log, and therefore no records will be captured by an Extract, until such time that the primary and secondary replicas are synchronized. See this article from Microsoft for more details:

  https://docs.microsoft.com/en-us/sql/database-engine/availability-groups/windows/replicate-track-change-data-capture-always-on-availability?view=sql-server-2017
- When running an Extract from a middle tier Windows or Linux server, set the middle tier server's date, time, and time zone to the same as that of the primary replica.
- Upon failover from a primary to a secondary replica, reinstall the Oracle GoldenGate CDC Cleanup job on the new primary by re-running the `ogg_cdc_cleanup_setup.bat` file with the `createJob` option.
- If Extract is configured to capture from a readable secondary replica, but not configured with read-only routing, the SQL Server CDC Capture job must be created against the secondary replica prior to starting the Extract, as the Extract will check if the job exists. To create the SQL Server CDC Capture job, any potential secondary that will have an Extract connected to it, must at some point be set to a writable Primary database and then follow the steps above, under supplemental logging, to manually add the SQL Server CDC Capture job.

- If uninstalling Oracle GoldenGate and disabling Change Data Capture on a database that is part of an Always On availability group, follow the extra steps provided in Disabling Change Data Capture.

11 CDC Capture Method Operational Considerations

This section provides information about the SQL Server CDC Capture options, features, and recommended settings.

Topics:

- Tuning SQL Server Change Data Capture
- Oracle GoldenGate CDC Object Versioning
- Valid and Invalid Extract Parameters for SQL Server Change Data Capture
- Details of the Oracle GoldenGate CDC Cleanup Process
- Changing from Classic Extract to a CDC Extract

11.1 Tuning SQL Server Change Data Capture

The following information is useful in improving the capture performance of the Extract.

- Ensure that **Auto Update Statistics** is enabled for the database. Updated statistics on the `cdc.OracleGG_#####_CT tables`, `cdc.lsn_time_mapping table`, and `OracleGGTranTables` table is crucial to the performance of the Extract. If **Auto Update Statistics** is disabled at the database level, then you can create a **SQL Server Agent** job that routinely updates the statistics for those objects.
- The SQL Server Change Data Capture job collects data from the SQL Server transaction log and loads it into the Change Data Capture staging tables within the database.

  As part of the job that is created, there are several available tuning parameters that can be used, and information on how to best tune the job can be found in the following article: https://technet.microsoft.com/en-us/library/dd266396(v=sql.100).aspx

  As a general recommendation, you should change the SQL Server Change Data Capture Job polling interval from the default of 5 seconds to 1 second.

To change the default polling interval of the CDC Capture job, execute the following queries against the database:

```
EXEC [sys].[sp_cdc_change_job]

@job_type = N'capture',

@pollinginterval = 1,

GO

–stops cdc job

EXEC [sys].[sp_cdc_stop_job]

@job_type = N'capture'

GO

–restarts cdc job for new polling interval to take affect

EXEC [sys].[sp_cdc_start_job]

@job_type = N'capture'
```

11.2 Oracle GoldenGate CDC Object Versioning

Oracle GoldenGate provides a version tracking subsystem to track the CDC objects that are created by Oracle GoldenGate when enabling supplemental logging. These objects are:

- Oracle GoldenGate change tracking tables in the format `OracleGG_`*`object id`*`_CT`.
- Stored procedures in the format *`fetch_database name_object id`*
- Stored procedures `OracleCDCExtract`, `OracleGGCreateProcs`, and `OracleGGCreateNextBatch`.
- After successfully completing the `ADD TRANDATA` command, GGSCI creates a table called `OracleGGVersion` under the `GGSCHEMA` specified in the GLOBALS file, if it does not already exist.

  Next, GGSCI inserts a record into the table that tracks the start and end time of the `TRANDATA` session. When Extract starts up, it checks for consistency between itself and the Oracle GoldenGate CDC objects by comparing its internal version

number with the version numbers found in the `OracleGGVersion` table. If it finds that the version numbers do not match, it abends with a message similar to the following:

```
ERROR OGG-05337 The Oracle GoldenGate CDC object versions on database,
source, are not consistent with the expected version, 2. The following
versions(s) were found: 1. Rerun ADD TRANDATA for all tables previously
enabled, including heartbeat, heartbeat seed, and filter tables.
```

11.3 Valid and Invalid Extract Parameters for SQL Server Change Data Capture

This section describes parameters used for the CDC Capture method. For more information about supported and unsupported parameters for the CDC Capture method, review *Reference for Oracle GoldenGate*.

`TRANLOGOPTIONS LOB_CHUNK_SIZE`

The Extract parameter `LOB_CHUNK_SIZE` is added for the CDC Capture method to support large objects. If you have huge LOB data sizes, then you can adjust the `LOB_CHUNK_SIZE` from the default of 4000 bytes, to a higher value up to 65535 bytes, so that the fetch size is increased, reducing the trips needed to fetch the entire LOB.

Example: `TRANLOGOPTIONS LOB_CHUNK_SIZE 8000`

`TRANLOGOPTIONS MANAGECDCCLEANUP/NOMANAGECDCCLEANUP`

The Extract parameter `MANAGECDCCLEANUP/NOMANAGECDCCLEANUP` is used by the CDC Capture method to instruct the Extract on whether or not to maintain recovery checkpoint data in the Oracle GoldenGate CDC Cleanup job. The default value is `MANAGECDCCLEANUP` and it doesn't have to be explicitly listed in the Extract. However, it does require creating the Oracle GoldenGate CDC Cleanup job prior to starting the Extract. `MANAGECDCCLEANUP` should be used for all production environments, where `NOMANAGECDCCLEANUP` may be used for temporary and testing implementations as needed.

Example: `TRANLOGOPTIONS MANAGECDCCLEANUP`

`TRANLOGOPTIONS EXCLUDEUSER/EXCLUDETRANS`

The SQL Server CDC Capture job does not capture user information or transaction names associated with a transaction, and as this information is not logged in the CDC staging tables, Extract has no method of excluding DML from a specific user or DML of a specific transaction name. The `EXCLUDEUSER` and `EXCLUDETRANS` parameters are therefore not valid for the CDC Capture process.

`TRANLOGOPTIONS MANAGESECONDARYTRUNCATIONPOINT/NOMANAGESECONDARYTRUNCATIONPOINT/ACTIVESECONDARYTRUNCATIONPOINT`

The SQL Server Change Data Capture job is the only process that captures data from the transaction log when using the Oracle GoldenGate CDC Capture method. The secondary truncation point management is not handled by the Extract, and for the Change Data Capture Extract, these parameters are not valid.

TRANLOGOPTIONS ALWAYSONREADONLYROUTING

The `ALWAYSONREADONLYROUTING` parameter allows Extract for SQL Server to route its read-only processing to an available read-intent Secondary when connected to an Always On availability group listener.

TRANLOGOPTIONS QUERYTIMEOUT

Specifies how long queries to SQL Server will wait for results before reporting a timeout error message. This option takes an integer value to represent the number of seconds. The default query timeout value is 300 seconds (5 minutes). The minimum value is 0 seconds (infinite timeout). The maximum is 2147483645 seconds.

TRANLOGOPTIONS TRANCOUNT

Allows adjustment of the number of transactions processed per each call by Extract to pull data from the SQL Server change data capture staging tables. Based on your transaction workload, adjusting this value may improve capture rate throughput, although not all workloads will be positively impacted. The minimum value is 1, maximum is 100, and the default is 10.

11.4 Details of the Oracle GoldenGate CDC Cleanup Process

The Oracle GoldenGate CDC Cleanup job is required for a CDC Extract by default, since Extract defaults to `TRANLOGOPTIONS MANAGECDCCLEANUP`. It is installed from a Windows batch file (`ogg_cdc_cleanup_setup.bat`), which uses `sqlcmd` to connect to the source SQL Server database and create the necessary objects and job.

There should be one job for each database enabled for CDC Capture, and you must create the job and objects following the steps mentioned in the Preparing the Database for Oracle GoldenGate — CDC Capture section of this document.

Additional options for the utility are discussed in the following sections.

The steps below require a SQL Server authenticated database user who is a member of the SQL Server System Administrators (`sysadmin`) role. Windows authentication is not supported for the `.bat` batch file.

Removing an Extract from the Database

When the Oracle GoldenGate CDC Cleanup object tables exist, each CDC Extract that is started against that database will create an entry in the `OracleGGExtractCheckpoint` table. This entry tracks a particular Extract's point in time recovery checkpoint, which is used as the cutoff LSN for the Oracle GoldenGate CDC cleanup tasks. If there are multiple Extracts running, each logging more recent recovery checkpoints in the table, but one Extract has been removed from the system without removing its entry into the `OracleGGExtractCheckpoint` table, then no data will be purged newer than that deleted Extract's old recovery checkpoint for all of the CDC staging tables. So when deleting an Extract from the database, follow the steps below to remove the Extract from the `OracleGGExtractCheckpoint` table if more than one Extract is running against the database.

1. Log in to the Database with `DBLOGIN` from GGSCI:

```
DBLOGIN SOURCEDB dsn_name USERIDALIAS alias_name
```

2. Stop the Extract:

```
STOP EXTRACT extract_name
```

3. Delete the Extract:

```
DELETE EXTRACT extract_name
```

By logging in to the database, the `DELETE EXTRACT` command removes the entry from the `OracleGGExtractCheckpoint` table, for that specific Extract.

Modifying the Oracle GoldenGate CDC Cleanup Job

The default schedule, retention period and operation batch size for the Oracle GoldenGate CDC Cleanup job of a database is to run every 10 minutes, with a data retention policy of 72 hours (listed as 4320 minutes), purging in batches of 500 records per transaction until the retention policy is meet, not to exceed the recovery checkpoint data of the Extract.

For variations in customer environments, or change data table data retention requirements, it may be necessary to adjust these properties to increase the purge batch size or to adjust retention policies and the job run-time schedule.

To adjust the job execution frequency, manually modify the schedule for the `OracleGGCleanup_dbname_Job` job within SQL Server Agent. If you need to adjust the retention period or purge batch size, you must manually edit the job step for the `OracleGGCleanup_dbname_Job` job within SQL Server Agent. The job step passes two parameters to the cleanup stored procedure, and you can modify the value for `@retention_minutes` to adjust the data retention policy as needed, or modify the `@threshold` value to increase or decrease the purge batch size. In high transactional environments, it may be necessary to increase the `@threshold` value to a number such as 10000. Monitoring the amount of time that it takes for the job to run within each cycle can be used to determine effective `@threshold` values.

Deleting the Oracle GoldenGate CDC Cleanup Job

If you no longer require the Oracle GoldenGate CDC Cleanup job and associated objects and need to remove them, perform the following steps:

1. Open a command prompt and change to the Oracle GoldenGate installation folder.

2. Run the `ogg_cdc_cleanup_setup.bat` file, providing the following variable values:

`ogg_cdc_cleanup_setup.bat dropJob` *`userid password databasename servername\instancename schema`*

Example: `ogg_cdc_cleanup_setup.bat dropJob ggsuser ggspword db1 server1\inst1 ogg`

11.5 Changing from Classic Extract to a CDC Extract

If you plan to change from using a Classic Extract from Oracle GoldenGate 12c (12.3.0.1) or earlier, to an Oracle GoldenGate 21c CDC Extract, then you must remove the supplemental logging that was implemented using the Classic Extract installation method, and re-enable supplemental logging using the CDC Extract installation binaries, as the calls to enable `TRANDATA` are different between the two versions, and the implementation of `TRANDATA` for Classic Extract is not supported by the CDC Extract.

Follow these general guidelines to remove and re-enable supplemental logging. Special consideration and planning should be involved if migrating from Classic to CDC Extract in a production system. The information provided here does not cover all requirements and is only offered as general requirements regarding supplemental logging:

1. Ensure that the Classic Extract has processed all remaining data in the logs and can be gracefully stopped.

2. Do one of the following, depending on how Extract was running in relation to other replication or CDC components:

- 1. If Extract was *not* running concurrently with SQL Server transactional replication or a non-Oracle CDC configuration on the same database, open a query session in Management Studio and issue the following statement against the source database to disable and delete any CDC or replication components, and to clear the secondary truncation point.

  ```
  EXEC sys.sp_cdc_disable_db
  ```

- 2. If Extract was running *concurrently* with SQL Server transactional replication or a non-Oracle CDC configuration on the same database, run GGSCI from the Classic Extract's installation folder, login to the source database with the `DBLOGIN`, and then issue the following command for each table that is in the Extract configuration. You can use a wildcard to specify multiple table names

  ```
  DELETE TRANDATA owner.table
  DELETE TRANDATA owner.*
  ```

3. Delete any heartbeat table entries if one was installed.

```
DELETE HEARTBEATTABLE
```

4. Using the Oracle GoldenGate CDC Extract installation binaries, follow the steps listed in Preparing the Database for Oracle GoldenGate — CDC Capture to re-enable supplemental logging and other necessary components, and re-add the heartbeat table.