Configuring Oracle GoldenGate for **Active-Active High Availability**

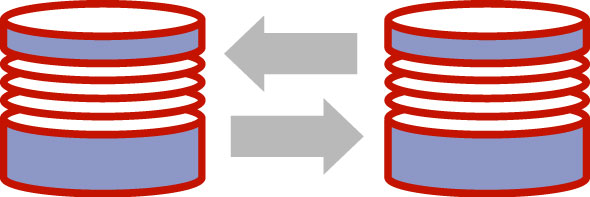
This chapter describes how to configure Oracle GoldenGate for active-active high availability.

This chapter includes the following sections:

* [Overview of an Active-active Configuration](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#i1046882)
* [Considerations for an Active-Active Configuration](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#BAGBFFJI)
* [Preventing Data Looping](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#i1036715)
* [Managing Conflicts](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#i1018918)
* [Additional Information](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#BAGCBJDG)
* [Creating an Active-Active Configuration](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#BAGBEEGG)

9.1 Overview of an Active-active Configuration

Oracle GoldenGate supports an active-active bi-directional configuration, where there are two systems with identical sets of data that can be changed by application users on either system. Oracle GoldenGate replicates transactional data changes from each database to the other to keep both sets of data current.

  
[Description of the illustration simple\_config\_bidirectional.jpg](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/img_text/simple_config_bidirectional.htm)

In a bi-directional configuration, there is a complete set of active Oracle GoldenGate processes on each system. Data captured by an Extract process on one system is propagated to the other system, where it is applied by a local Replicat process.

This configuration supports load sharing. It can be used for disaster tolerance if the business applications are identical on any two peers. Bidirectional synchronization is supported for all database types that are supported by Oracle GoldenGate.

Oracle GoldenGate supports active-active configurations for:

* DB2 on z/OS, LUW, and IBM i
* MySQL
* Oracle
* SQL/MX
* SQL Server
* Sybase
* Teradata

Oracle GoldenGate supports DDL replication in an Oracle active-active configuration. DDL support is available for Oracle and Teradata databases.

9.2 Considerations for an **Active-Active** Configuration

The following considerations apply in an active-active configuration. In addition, review the Oracle GoldenGate installation and configuration document for your type of database to see if there are any other limitations or requirements to support a bi-directional configuration.

**PS: Een issue wat hier nog niet genoemd is, is het gebruik van meerdere databases/schema’s die in SYNC moeten blijven. Onze INTERSPEC moet consistent zijn met onze UNILAB database. Hoe zorgen we er voor dat mutaties van beide databases syncrhoon in juiste volgorde worden ge-repliceerd?**

**PS: Het zou misschien veel eenvoudiger zijn als we maar 1 DB zouden hebben. Is het qua objecten toch al mogelijk dat we alle SCHEMA’s van INTERSPEC en UNILAB in 1 database op zouden nemen? Is het onderzoeken waard, misschien ook navragen bij ATS/SIEMENS.**

**PS: Qua gebruik / raadplegen van objecten kan het wel zijn dat vanuit verschillende locaties een zelfde SPEC wordt geraadpleegd/gewijzigd. Als sequence-gebruik etc. overal goed geimplementeerd is zou dat niet uit moeten maken. Blijft alleen toch wel een risico voor CONFLICTEN, omdat we op logisch niveau geen lock kunnen zetten. Theoretisch zou eenzelfde SPEC vanuit 2 verschillende locaties gewijzigd kunnen worden, omdat er technisch niet op hetzelfde RECORD een oracle-lock komt te staan. Dan komen in het stukje conflict-hantering terecht.**

9.2.1 TRUNCATES

**Bi-directional replication of TRUNCATES is not supported**, but you can configure these operations to be replicated in one direction, while data is replicated in both directions. To replicate TRUNCATES (if supported by Oracle GoldenGate for the database) in an active-active configuration, the TRUNCATES must originate only from one database, and only from the same database each time.

Configure the environment as follows:

* Configure all database roles so that they cannot execute TRUNCATE from any database other than the one that is designated for this purpose.
* On the system where TRUNCATE will be permitted, configure the Extract and Replicat parameter files to contain the GETTRUNCATES parameter.
* On the other system, configure the Extract and Replicat parameter files to contain the IGNORETRUNCATES parameter. No TRUNCATES should be performed on this system by applications that are part of the Oracle GoldenGate configuration.

**PS: weet niet zeker of onze db-schonings-jobs soms een truncate gebruiken. Nakijken.**

9.2.2 Application Design

**When using Active-Active replication, the time zones must be the same on both systems so that timestamp-based conflict resolution and detection can operate**.

Active-active replication is not recommended for use with commercially available packaged business applications, unless the application is designed to support it. Among the obstacles that these applications present are:

* Packaged applications might contain objects and data types that are not supported by Oracle GoldenGate.
* They might perform automatic DML operations that you cannot control, but which will be replicated by Oracle GoldenGate and cause conflicts when applied by Replicat.
* You probably cannot control the data structures to make modifications that are required for active-active replication.

**PS: Dit moeten we nader uitzoeken. Het is volgens mij niet zo dat we in zelfde time-zones zitten, maar weet niet zeker hoe ze in de database opgeslagen worden. Als dat allemaal in UTC-tijden zou zijn en de timezone-info alleen voor presentatie wordt gebruikt dan hoeven we misschien geen probleem te hebben.**

9.2.3 Keys

**For accurate detection of conflicts, all records must have a unique, not-null identifier**. If possible, create a primary key. If that is not possible, use a unique key or create a substitute key with a KEYCOLS option of the MAP and TABLE parameters. In the absence of a unique identifier, Oracle GoldenGate uses all of the columns that are valid in a WHERE clause, but this will degrade performance if the table contains numerous columns.

To maintain data integrity and prevent errors, the following must be true of the key that you use for any given table:

* contain the same columns in all of the databases where that table resides.
* contain the same values in each set of corresponding rows across the databases.

**PS: Er zijn bij ons tabellen waarbij PK ook null-columns kan bevatten. Wat gebeurt er als een PK kan wijzigt, dan gebeurt bij ons nog wel eens. Zou eigenlijk toch een PK op moeten komen, en een UK voor huidige pk-kolommen.**

9.2.4 Triggers and Cascaded Deletes

Triggers and ON DELETE CASCADE constraints generate DML operations that can be replicated by Oracle GoldenGate. To prevent the local DML from conflicting with the replicated DML from these operations, do the following:

* **Modify triggers to ignore DML operations that are applied by Replicat**. If the target is an Oracle database, Replicat handles triggers without any additional configuration when in integrated mode. Parameter options are available for a nonintegrated Replicat for Oracle. See [*Installing and Configuring Oracle GoldenGate for Oracle Database*](https://docs.oracle.com/goldengate/1212/gg-winux/GIORA/nonint_replicat.htm#GIORA359).

**PS: Dat komt bij ons wel voor, moeten we dus iets voor verzinnen. Aan de hand waarvan kunnen we zien of het een DML vanuit de applicatie is, of vanuit GOLDENGATE ?   
De vraag is ook hoe we omgaan met onze EVENTS. We willen niet dat een EVENT en als event wordt verwerkt, en daarnaast ook nog eens via normale DML op de tabellen wordt verwerkt…**

* Disable ON DELETE CASCADE constraints and use a trigger on the parent table to perform the required delete(s) to the child tables. Create it as a BEFORE trigger so that the child tables are deleted before the delete operation is performed on the parent table. This reverses the logical order of a cascaded delete but is necessary so that the operations are replicated in the correct order to prevent "table not found" errors on the target. (Note: For Oracle targets, if Replicat is in integrated mode, constraints are handled automatically without special configuration.)

**PS: Dit soort constructies ben ik “gelukkig” niet tegengekomen in INTERSPEC/UNILAB.**

**Note:**

IDENTITY columns cannot be used with bidirectional configurations for Sybase. See other IDENTITY limitations for SQL Server in [*Installing and Configuring Oracle GoldenGate for SQL Server*](https://docs.oracle.com/goldengate/1212/gg-winux/GIMSS/toc.htm).

9.2.5 Database-Generated Values

**Do not replicate database-generated sequential values, such as Oracle sequences, in a bi-directional configuration**. **The range of values must be different on each system, with no chance of overlap**. For example, in a two-database environment, you can have one server generate even values, and the other odd. For an *n*-server environment, start each key at a different value and increment the values by the number of servers in the environment. This method may not be available to all types of applications or databases. If the application permits, you can add a location identifier to the value to enforce uniqueness.

PS: We gebruiken wel (mn. In INTERSPEC) heel veel sequences. Daar zouden we dan per PLANT onderscheid in moeten gaan maken. Dan moeten de sequences wel uitsluiten voor replicatie, en deze vanuit de triggers niet gebruiken zonder extra conditie met check of een ID met sequence-waarde al gevuld is.

9.2.6 Database Configuration

One of the databases must be designated as the *trusted source*. This is the primary database and its host system from which the other database is derived in the initial synchronization phase and in any subsequent resynchronizations that become necessary. Maintain frequent backups of the trusted source data.

9.3 Preventing Data Looping

In a bidirectional configuration, SQL changes that are replicated from one system to another must be prevented from being replicated back to the first system. Otherwise, it moves back and forth in an endless loop, as in this example:

1. A user application updates a row on system A.
2. Extract extracts the row on system A and sends it to system B.
3. Replicat updates the row on system B.
4. Extract extracts the row on system B and sends it back to system A.
5. The row is applied on system A (for the second time).
6. This loop continues endlessly.

**To prevent data loopback, you may need to provide instructions that**:

* prevent the capture of SQL operations that are generated by Replicat, but enable the capture of SQL operations that are generated by business applications if they contain objects that are specified in the Extract parameter file.
* identify local Replicat transactions, in order for the Extract process to ignore them.

**PS: Is er bekend (zijn er voorbeelden) van HOE we dit onderscheid zouden moeten kunnen maken?  
(O, zie hieronder 😊)**

9.3.1 Preventing the Capture of Replicat Operations

Depending on which database you are using, you may or may not need to provide explicit instructions to prevent the capture of Replicat operations.

9.3.1.1 Preventing the Capture of Replicat Transactions (Oracle)

To prevent the capture of SQL that is applied by Replicat to an Oracle database, there are different options depending on the Extract capture mode:

* When Extract is in classic or integrated capture mode, use the TRANLOGOPTIONS parameter with the EXCLUDETAG *tag* option. This parameter directs the Extract process to ignore transactions that are tagged with the specified redo tag. See [Section 9.3.2](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#BAGBBBCC) to set the tag value.
* When Extract is in classic capture mode, use the Extract TRANLOGOPTIONS parameter with the EXCLUDEUSER or EXCLUDEUSERID option to exclude the user name or ID that is used by Replicat to apply transactions. Multiple EXCLUDEUSER statements can be used. The specified user is subject to the rules of the GETREPLICATES or IGNOREREPLICATES parameter. See [Section 9.3.1.3](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#BAGBGHAD) for more information.

**PS: wat is voorstel van oracle om voor een welke manier van REPLICEREN te kiezen (classic?). Dan zullen we met EXCLUDETAG een TAG-versie moeten zetten, of met EXCLUDEUSER een overzicht van users moeten maken. Of kunnen we een PLANT-ID voor een LOCATIE gebruiken, dat zou veel logischer zijn. Qua infra weet je vanuit welke locatie de mutaties komen, of niet?**

9.3.1.2 Preventing Capture of Replicat Transactions (Teradata)

To prevent the capture of SQL that is applied by Replicat to a Teradata database, set the Replicat session to override Teradata replication. Use the following SQLEXEC statements at the root level of the Replicat parameter file:

SQLEXEC 'SET SESSION OVERRIDE REPLICATION ON;'

SQLEXEC 'COMMIT;'

These SQLEXEC statements execute a procedure that sets the Replicat session automatically at startup.

9.3.1.3 Preventing Capture of Replicat Transactions (Other Databases)

To prevent the capture of SQL that is applied by Replicat to other database types (including Oracle, if Extract operates in classic capture mode), use the following parameters:

* GETAPPLOPS | IGNOREAPPLOPS: Controls whether or not data operations (DML) produced by business applications *except Replicat* are included in the content that Extract writes to a specific trail or file.
* GETREPLICATES | IGNOREREPLICATES: Controls whether or not DML operations produced by *Replicat* are included in the content that Extract writes to a specific trail or file.

9.3.2 Identifying Replicat Transactions

To configure Extract to identify Replicat transactions, follow the instructions for the database from which Extract will capture data.

9.3.2.1 DB2 on z/OS, LUW, IBM i, and Informix

Identify the Replicat user name by using the following parameter statement in the Extract parameter file.

TRANLOGOPTIONS EXCLUDEUSER *user*

This parameter statement marks all data transactions that are generated by this user as Replicat transactions. The user name is included in the transaction record that is read by Extract.

**Note:**

With Informix, Oracle GoldenGate Replicat processes always perform a DELETE operation on the target database when the source operation (in the trail) is TRUNCATE.

9.3.2.2 MySQL and NonStop SQL/MX

Identify the name of the Replicat checkpoint table by using the following parameter statement in the Extract parameter file.

TRANLOGOPTIONS FILTERTABLE *table\_name*

Replicat writes a checkpoint to the checkpoint table at the end of each of its transactions as part of its checkpoint procedure. (This is the table that is created with the ADD CHECKPOINTTABLE command.) Because every Replicat transaction includes a write to this table, it can be used to identify Replicat transactions in a bi-directional configuration. FILTERTABLE identifies the name of the checkpoint table, so that Extract ignores transactions that contain any operations on it.

**Note:**

PURGEDATA is not supported for NonStop SQL/MX in a bidirectional configuration. Because PURGEDATA/TRUNCATE operations are DDL, they are implicit transactions, so Oracle GoldenGate cannot update the checkpoint table within that transaction.

9.3.2.3 Oracle

There are multiple ways to identify Replicat transaction in an Oracle environment. When Replicat is in classic or integrated mode, you use the following parameters:

* Use DBOPTIONS with the SETTAG option in the Replicat parameter file. Replicat tags the transactions being applied with the specified value, which identifies those transactions in the redo stream. The default SETTAG value is 00. Valid values are a single TAG value consisting of hexadecimal digits. For more information about tags, see [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/toc.htm).
* Use the TRANLOGOPTIONS parameter with the EXCLUDETAG option in the Extract parameter file. The logmining server associated with that Extract excludes redo that is tagged with the SETTAG value.

The following shows how SETTAG can be set in the Replicat parameter file:

DBOPTIONS SETTAG 0935

The following shows how EXCLUDETAG can be set in the Extract parameter file:

TRANLOGOPTIONS EXCLUDETAG 0935

If you are excluding multiple tags, each must have a separate TRANLOGOPTIONS EXCLUDETAG statement specified.

You can also use the transaction name or userid of the Replicat user to identify Replicat transactions. You can choose which of these to ignore when you configure Extract. See [Section 9.3.1.1](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#BAGBJGHG).

For more information, see [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/gg_parameters174.htm#GWURF712).

9.3.2.4 SQL Server

Identify the Replicat transaction name by using the following parameter statement in the Extract parameter file.

TRANLOGOPTIONS EXCLUDETRANS *transaction\_name*

This parameter statement is only required if the Replicat transaction name is set to something other than the default of ggs\_repl.

9.3.2.5 Sybase

Do any of the following:

* Identify a Replicat transaction name by using the following parameter statement in the Extract parameter file.
* TRANLOGOPTIONS EXCLUDETRANS *transaction name*
* Identify the Replicat user name by using the following parameter statement in the Extract parameter file.
* TRANLOGOPTIONS EXCLUDEUSER *user name*

EXCLUDEUSER marks all transactions generated by this user as Replicat transactions. The user name is included in the transaction record that is read by Extract.

* Do nothing and allow Replicat to use the default transaction name of ggs\_repl.

9.3.2.6 Teradata

You do not need to identify Replicat transactions that are applied to a Teradata database.

9.3.3 Replicating DDL in a Bi-directional Configuration

Additional consideration must be taken when replicating DDL, currently supported for Oracle and Teradata. For more information, see the following:

* [*Installing and Configuring Oracle GoldenGate for Oracle Database*](https://docs.oracle.com/goldengate/1212/gg-winux/GIORA/ddl.htm#GIORA341)
* *Installing and Configuring Oracle GoldenGate for Teradata*

9.4 Managing Conflicts

Uniform conflict-resolution procedures must be in place on all systems in an active-active configuration. Conflicts should be identified immediately and handled with as much automation as possible; however, different business applications will present their own unique set of requirements in this area.

Because Oracle GoldenGate is an asynchronous solution, conflicts can occur when modifications are made to identical sets of data on separate systems at (or almost at) the same time. Conflicts occur when the timing of simultaneous changes results in one of these out-of-sync conditions:

* A **uniqueness conflict** occurs when Replicat applies an insert or update operation that violates a uniqueness integrity constraint, such as a PRIMARY KEY or UNIQUE constraint. An example of this conflict type is when two transactions originate from two different databases, and each one inserts a row into a table with the same primary key value.
* An **update conflict** occurs when Replicat applies an update that conflicts with another update to the same row. Update conflicts happen when two transactions that originate from different databases update the same row at nearly the same time. Replicat detects an update conflict when there is a difference between the old values (the before values) that are stored in the trail record and the current values of the same row in the target database.
* A **delete conflict** occurs when two transactions originate at different databases, and one deletes a row while the other updates or deletes the same row. In this case, the row does not exist to be either updated or deleted. Replicat cannot find the row because the primary key does not exist.

For example, UserA on DatabaseA updates a row, and UserB on DatabaseB updates the same row. If UserB's transaction occurs before UserA's transaction is synchronized to DatabaseB, there will be a conflict on the replicated transaction.

A more complicated example involves three databases and illustrates a more complex ordering conflict. Assume three databases A, B, and C. Suppose a user inserts a row at database A, which is then replicated to database B. Another user then modifies the row at database B, and the row modification is replicated to database C. If the row modification from B arrives at database C before the row insert from database A, C will detect a conflict.

Where possible, try to minimize or eliminate any chance of conflict. Some ways to do so are:

* Configure the applications to restrict which columns can be modified in each database. For example, you could limit access based on geographical area, such as by allowing different sales regions to modify only the records of their own customers. As another example, you could allow a customer service application on one database to modify only the NAME and ADDRESS columns of a customer table, while allowing a financial application on another database to modify only the BALANCE column. In each of those cases, there cannot be a conflict caused by concurrent updates to the same record.
* Keep synchronization latency low. If UserA on DatabaseA and UserB on DatabaseB both update the same rows at about the same time, and UserA's transaction gets replicated to the target row before UserB's transaction is completed, conflict is avoided. See [Chapter 19, "Tuning the Performance of Oracle GoldenGate"](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_performance.htm#g998643) for suggestions on improving the performance of the Oracle GoldenGate processes.

To avoid conflicts, replication latency must be kept as low as possible. When conflicts are unavoidable, they must be identified immediately and resolved with as much automation as possible, either through the Oracle GoldenGate Conflict Detection and Resolution (CDR) feature, or through methods developed on your own. Custom methods can be integrated into Oracle GoldenGate processing through the SQLEXEC and user exit functionality. See [Chapter 10, "Configuring Conflict Detection and Resolution"](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/conflict_resolution.htm#CJHFCDHF) for more information about using Oracle GoldenGate to handle conflicts.

9.5 Additional Information

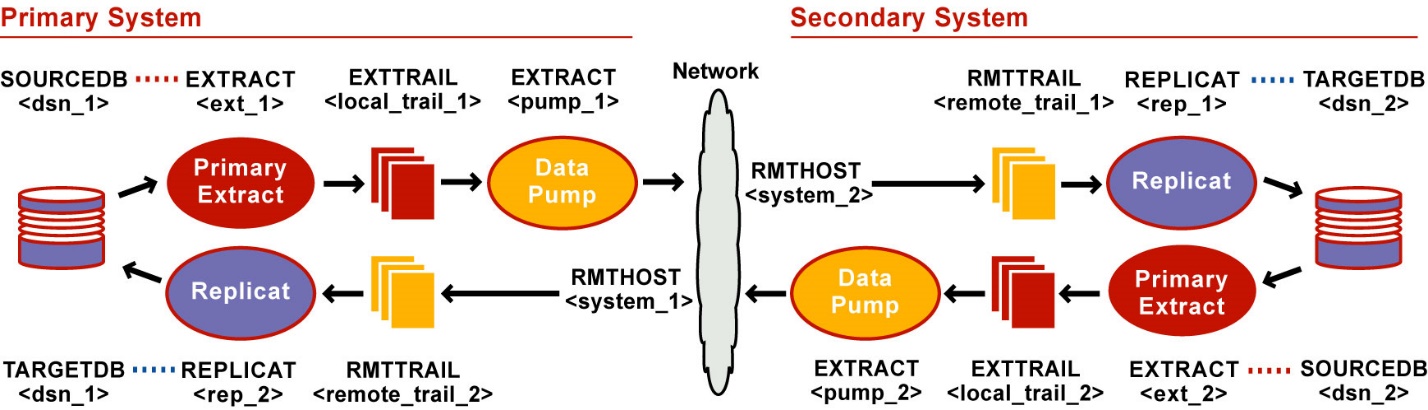
The following documentation provides additional information of relevance to configuring Oracle GoldenGate.

* For additional system requirements, process configuration, and database setup requirements, see the Oracle GoldenGate installation and configuration document for your database type. These guides are listed under ["Related Documents"](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/preface.htm#CHDCIJJH) in the [Preface](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/preface.htm#CIADAFBH) of this book.
* For detailed instructions on configuring Oracle GoldenGate change capture and delivery groups, see [Chapter 14, "Configuring Online Change Synchronization."](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_changesync.htm#i999929)
* For additional tuning options for Oracle GoldenGate, see [Chapter 19, "Tuning the Performance of Oracle GoldenGate."](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_performance.htm#g998643)
* For complete syntax and descriptions of the Oracle GoldenGate commands and parameters, see [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/toc.htm).

9.6 Creating an Active-Active Configuration

Refer to [Figure 9-1](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_bidirectional.htm#BAGEBIAA) for a visual representation of the objects you will be creating.

***Figure 9-1 Oracle GoldenGate Configuration for Active-active Synchronization***

  
[Description of "Figure 9-1 Oracle GoldenGate Configuration for Active-active Synchronization"](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/img_text/bidirectional_config.htm)

9.6.1 Prerequisites on Both Systems

Perform these prerequisite tasks on both systems:

1. Create a Replicat checkpoint table (unless using Oracle integrated Replicat). See [Section 14.3, "Creating a Checkpoint Table"](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_changesync.htm#i1041277) for instructions.
2. Configure the Manager process. See [Chapter 3, "Configuring Manager and Network Communications"](https://docs.oracle.com/goldengate/1212/gg-winux/GWUAD/wu_manager.htm#i1033185) for instructions.

9.6.2 Configuration from Primary System to Secondary System

These steps add the processes necessary to send data from the primary system to the secondary database.

**To configure the primary Extract group**

Perform these steps on the primary system.

1. Use the ADD EXTRACT command to create a primary Extract group. For documentation purposes, this group is called *ext\_1*.
2. ADD EXTRACT *ext\_1*, {TRANLOG | INTEGRATED TRANLOG}, BEGIN *time*

See [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/ggsci_commands006.htm#GWURF122) for detailed information about these and other ADD EXTRACT options that may be required for your installation.

1. Use the ADD EXTTRAIL command to add a local trail. For documentation purposes, this trail is called *local\_trail\_1*.
2. ADD EXTTRAIL *local\_trail\_1*, EXTRACT *ext\_1*

For EXTRACT, specify the *ext\_1* group to write to this trail

1. Use the EDIT PARAMS command to create a parameter file for the *ext\_1* group. Include the following parameters plus any others that apply to your database environment. For possible additional required parameters, see the Oracle GoldenGate installation and setup guide for your database.
2. -- Identify the Extract group:
3. EXTRACT *ext\_1*
4. -- Specify database login information as needed for the database:
5. [SOURCEDB *dsn\_1*][, USERIDALIAS *alias*]
6. -- Specify the local trail that this Extract writes to
7. -- and the encryption algorithm:
8. ENCRYPTTRAIL *algorithm*
9. EXTTRAIL *local\_trail\_1*
10. -- Exclude Replicat transactions. Uncomment ONE of the following:
11. -- DB2 z/OS, DB2 LUW, DB2 IBM i, Oracle (classic capture), and
12. -- Sybase:
13. -- TRANLOGOPTIONS EXCLUDEUSER *Replicat\_user*
14. -- Oracle (classic capture) alternative to EXCLUDEUSER:
15. -- EXCLUDEUSERID *Oracle\_uid*
16. -- Oracle integrated capture:
17. -- EXCLUDETAG *tag*
18. -- SQL Server and Sybase:
19. -- TRANLOGOPTIONS EXCLUDETRANS *transaction\_name*
20. -- SQL/MX:
21. -- TRANLOGOPTIONS FILTERTABLE *checkpoint\_table\_name*
22. -- Teradata:
23. -- SQLEXEC 'SET SESSION OVERRIDE REPLICATION ON;'
24. -- SQLEXEC 'COMMIT;'
25. -- Specify API commands if Teradata:
26. VAM library name, PARAMS ('*param*' [, '*param*'] [, ...])
27. -- Capture before images for conflict resolution:
28. GETBEFORECOLS (ON *operation* {ALL | KEY | KEYINCLUDING (*col\_list*) | ALLEXCLUDING (*col\_list*)})
29. -- Log all scheduling columns for CDR and if using integrated Replicat
30. LOGALLSUPCOLS
31. -- Specify tables to be captured and (optional) columns to fetch:
32. TABLE [*container*.|*catalog*.]*owner*.\* [, FETCHCOLS *cols* | FETCHCOLSEXCEPT *cols*];

**To configure the data pump**

Perform these steps on the primary system.

1. Use the ADD EXTRACT command to create a data pump group. For documentation purposes, this group is called *pump\_1*.
2. ADD EXTRACT *pump\_1*, EXTTRAILSOURCE *local\_trail\_1*, BEGIN *time*

For EXTTRAILSOURCE, specify *local\_trail\_1* as the data source.

1. Use the ADD RMTTRAIL command to add a remote trail that will be created on the secondary system. For documentation purposes, this trail is called *remote\_trail\_1*.
2. ADD RMTTRAIL *remote\_trail\_1*, EXTRACT *pump\_1*

For EXTRACT, specify the *pump\_1* data pump to write to this trail.

See [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/ggsci_commands049.htm#GWURF227) for additional ADD RMTTRAIL options.

1. Use the EDIT PARAMS command to create a parameter file for the *pump\_1* group. Include the following parameters plus any others that apply to your database environment.
2. -- Identify the data pump group:
3. EXTRACT *pump\_1*
4. -- Specify database login information as needed for the database:
5. [SOURCEDB *dsn\_1*][, USERIDALIAS *alias*]
6. -- Decrypt the data only if the data pump must process it.
7. -- DECRYPTTRAIL
8. -- Specify the name or IP address of the secondary system
9. -- and optional encryption of data over TCP/IP:
10. RMTHOSTOPTIONS *system\_2*, MGRPORT *port\_number*, ENCRYPT *encryption\_options*
11. -- Specify remote trail and encryption algorithm on secondary system:
12. ENCRYPTTRAIL *algorithm*
13. RMTTRAIL *remote\_trail\_1*
14. -- Pass data through without mapping, filtering, conversion:
15. PASSTHRU
16. -- Specify tables to be captured:
17. TABLE [*container*.|*catalog*.]*owner*.\*;

**Note:**

Because data structures are usually identical in a bi-directional configuration, PASSTHRU mode improves performance.

**To configure the Replicat group**

Perform these steps on the secondary system.

1. Use the ADD REPLICAT command to create a Replicat group. For documentation purposes, this group is called *rep\_1*.
2. ADD REPLICAT *rep\_1*
3. [, INTEGRATED | COORDINATED [MAXTHREADS *number*]]
4. , EXTTRAIL *remote\_trail\_1*, BEGIN *time*

For EXTTRAIL, specify *remote\_trail\_1* as the trail that this Replicat reads.

See [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/ggsci_commands020.htm#GWURF183) for detailed information about these and other options that may be required for your installation.

1. Use the EDIT PARAMS command to create a parameter file for the *rep\_1* group. Include the following parameters plus any others that apply to your database environment. For possible additional required parameters, see the Oracle GoldenGate installation and setup guide for your database.
2. -- Identify the Replicat group:
3. REPLICAT *rep\_1*
4. -- State that source and target definitions are identical:
5. ASSUMETARGETDEFS
6. -- Specify database login information as needed for the database:
7. [TARGETDB *dsn\_2*][, USERIDALIAS *alias*]
8. -- Specify error handling rules:
9. REPERROR (*error*, *response*)
10. -- Set non-default Replicat redo tag (if Extract is in integrated mode)
11. -- Default is 00.
12. SETTAG *tag\_value*
13. -- Specify tables for delivery, threads if coordinated Replicat
14. -- and conflict-resolution:
15. MAP [*container*.|*catalog*.]*owner*.\*, TARGET *owner*.\*, COMPARECOLS (ON *operation* {ALL | KEY | KEYINCLUDING (*col\_list*) | ALLEXCLUDING (*col\_list*)}), RESOLVECONFLICT (*conflict type* (*resolution\_name*, *resolution\_type* COLS (*col*[,...]))
16. [, THREAD (*thread\_ID*)]
17. [, THREADRANGE (*thread\_range*[, *column\_list*])]
18. ;
19. -- Specify mapping of exceptions to exceptions table:
20. MAP [*container*.|*catalog*.]*owner*.\*, TARGET *owner*.*exceptions*, EXCEPTIONSONLY;

9.6.3 Configuration from Secondary System to Primary System

These steps add the processes necessary to send data from the secondary system to the primary database.

**To configure the primary Extract group**

Perform these steps on the secondary system.

**Note:**

This is a reverse image of the configuration that you just created.

1. Use the ADD EXTRACT command to create a primary Extract group. For documentation purposes, this group is called *ext\_2*.
2. ADD EXTRACT *ext\_2*, {TRANLOG | INTEGRATED TRANLOG}, BEGIN *time*

See [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/ggsci_commands006.htm#GWURF122) for detailed information about these and other ADD EXTRACT options that may be required for your installation.

1. Use the ADD EXTTRAIL command to add a local trail. For documentation purposes, this trail is called *local\_trail\_2*.
2. ADD EXTTRAIL *local\_trail\_2*, EXTRACT *ext\_2*

For EXTRACT, specify the *ext\_2* group to write to this trail.

1. Use the EDIT PARAMS command to create a parameter file for the *ext\_2* group. Include the following parameters plus any others that apply to your database environment. For possible additional required parameters, see the Oracle GoldenGate installation and setup guide for your database.
2. -- Identify the Extract group:
3. EXTRACT *ext\_2*
4. -- Specify database login information as needed for the database:
5. [SOURCEDB *dsn\_2*][, USERIDALIAS *alias*]
6. -- Specify the local trail that this Extract writes to
7. -- and the encryption algorithm:
8. ENCRYPTTRAIL *algorithm*
9. EXTTRAIL *local\_trail\_2*
10. -- Exclude Replicat transactions. Uncomment ONE of the following:
11. -- DB2 z/OS, DB2 LUW, DB2 IBM i, Oracle, and Sybase:
12. -- TRANLOGOPTIONS EXCLUDEUSER *Replicat\_user*
13. -- Oracle alternative to EXCLUDEUSER:
14. -- EXCLUDEUSERID *Oracle\_uid*
15. -- Oracle integrated capture:
16. -- EXCLUDETAG *tag*
17. -- SQL Server and Sybase:
18. -- TRANLOGOPTIONS EXCLUDETRANS *transaction\_name*
19. -- SQL/MX:
20. -- TRANLOGOPTIONS FILTERTABLE *checkpoint\_table\_name*
21. -- Teradata:
22. -- SQLEXEC 'SET SESSION OVERRIDE REPLICATION ON;'
23. -- SQLEXEC 'COMMIT;'
24. -- Oracle:
25. -- TRACETABLE *trace\_table\_name*
26. -- Log all scheduling columns for CDR and if using integrated Replicat
27. LOGALLSUPCOLS
28. -- Capture before images for conflict resolution:
29. GETBEFORECOLS (ON *operation* {ALL | KEY | KEYINCLUDING (*col\_list*) | ALLEXCLUDING (*col\_list*)})
30. -- Specify tables to be captured and (optional) columns to fetch:
31. TABLE [*container*.|*catalog*.]*owner*.\* [, FETCHCOLS *cols* | FETCHCOLSEXCEPT *cols*];

**Note:**

To replicate Oracle DBFS data, specify the internally generated local *read-write* DBFS tables in the TABLE statement on each node. For more information on identifying these tables and configuring DBFS for propagation by Oracle GoldenGate, see [*Installing and Configuring Oracle GoldenGate for Oracle Database*](https://docs.oracle.com/goldengate/1212/gg-winux/GIORA/dbfs.htm#GIORA515).

**To configure the data pump**

Perform these steps on the secondary system.

1. Use the ADD EXTRACT command to create a data pump group. For documentation purposes, this group is called *pump\_2*.
2. ADD EXTRACT *pump\_2*, EXTTRAILSOURCE *local\_trail\_2*, BEGIN *time*

For EXTTRAILSOURCE, specify *local\_trail\_2* as the data source.

1. Use the ADD RMTTRAIL command to add a remote trail that will be created on the primary system. For documentation purposes, this trail is called *remote\_trail\_2*.
2. ADD RMTTRAIL *remote\_trail\_2*, EXTRACT *pump\_2*

For EXTRACT, specify the *pump\_2* data pump to write to this trail.

See [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/ggsci_commands049.htm#GWURF227) for additional ADD RMTTRAIL options.

1. Use the EDIT PARAMS command to create a parameter file for the *pump\_2* group. Include the following parameters plus any others that apply to your database environment.
2. -- Identify the data pump group:
3. EXTRACT *pump\_2*
4. -- Specify database login information as needed for the database:
5. [SOURCEDB *dsn\_2*][, USERIDALIAS *alias*]
6. -- Decrypt the data only if the data pump must process it.
7. -- DECRYPTTRAIL
8. -- Specify the name or IP address of the primary system
9. -- and optional encryption of data over TCP/IP:
10. RMTHOSTOPTIONS *system\_1*, MGRPORT *port\_number*, ENCRYPT *encryption\_options*
11. -- Specify the remote trail and encryption algorithm on the primary system:
12. ENCRYPTTRAIL *algorithm*
13. RMTTRAIL *remote\_trail\_2*
14. -- Pass data through without mapping, filtering, conversion:
15. PASSTHRU
16. -- Specify tables to be captured:
17. TABLE [*container*.|*catalog*.]*owner*.\*;

**Note:**

To replicate Oracle DBFS data, specify the internally generated local *read-write* DBFS tables in the TABLE statement on each node. For more information on identifying these tables and configuring DBFS for propagation by Oracle GoldenGate, see [*Installing and Configuring Oracle GoldenGate for Oracle Database*](https://docs.oracle.com/goldengate/1212/gg-winux/GIORA/dbfs.htm#GIORA515).

**To configure the Replicat group**

Perform these steps on the primary system.

1. Use the ADD REPLICAT command to create a Replicat group. For documentation purposes, this group is called *rep\_2*.
2. ADD REPLICAT *rep\_2*
3. [, INTEGRATED | COORDINATED [MAXTHREADS *number*]]
4. , EXTTRAIL *remote\_trail\_2*, BEGIN *time*

For EXTTRAIL, specify *remote\_trail\_2* as the trail that this Replicat reads.

See [*Reference for Oracle GoldenGate for Windows and UNIX*](https://docs.oracle.com/goldengate/1212/gg-winux/GWURF/ggsci_commands020.htm#GWURF183) for detailed information about these and other options that may be required for your installation.

1. Use the EDIT PARAMS command to create a parameter file for the *rep\_2* group. Include the following parameters plus any others that apply to your database environment. For possible additional required parameters, see the Oracle GoldenGate installation and setup guide for your database.
2. -- Identify the Replicat group:
3. REPLICAT *rep\_2*
4. -- State that source and target definitions are identical:
5. ASSUMETARGETDEFS
6. -- Specify database login information as needed for the database:
7. [TARGETDB *dsn\_1*][, USERIDALIAS *alias*]
8. -- Specify error handling rules:
9. REPERROR (*error*, *response*)
10. -- Specify tables for delivery, threads if coordinated Replicat
11. -- and conflict-resolution:
12. MAP [*container*.|*catalog*.]*owner*.\*, TARGET *owner*.\*, COMPARECOLS (ON *operation* {ALL | KEY | KEYINCLUDING (*col\_list*) | ALLEXCLUDING (*col\_list*)}), RESOLVECONFLICT (*conflict type* (*resolution\_name*, *resolution\_type* COLS (*col*[,...]))
13. [, THREAD (*thread\_ID*)]
14. [, THREADRANGE (*thread\_range*[, *column\_list*])]
15. ;
16. -- Specify mapping of exceptions to exceptions table:
17. MAP [*container*.|*catalog*.]*owner*.\*, TARGET *owner*.*exceptions*, EXCEPTIONSONLY;

**Note:**

To replicate Oracle DBFS data, specify the internally generated local *read-write* DBFS tables in the TABLE statement on each node. For more information on identifying these tables and configuring DBFS for propagation by Oracle GoldenGate, see [*Installing and Configuring Oracle GoldenGate for Oracle Database*](https://docs.oracle.com/goldengate/1212/gg-winux/GIORA/dbfs.htm#GIORA515).