Oracle® GoldenGate

Veridata 23c (23.1.0.0.0)

F85008-02
January 2025

Oracle GoldenGate Veridata 23c (23.1.0.0.0),

F85008-02

Contents

7 Secure

Preface

This article explains the installation and configuration of Oracle GoldenGate Veridata, the Veridata Web User Interface and all its command line utilities.

Audience

This article is intended for system administrators and database users to learn about Oracle GoldenGate Veridata. It is assumed that readers are familiar with web technologies and have a general understanding of Windows and UNIX platforms.

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.

Accessible Access to Oracle Support

Oracle customers who 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

- Oracle GoldenGate Documentation
- Oracle GoldenGate Veridata

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.

Convention	Meaning
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1
Overview

1.1 What is Oracle GoldenGate Veridata?

Oracle GoldenGate Veridata is a high-speed, data-comparison and repair solution that identifies, reports on, and fixes data discrepancies between heterogeneous databases without interrupting ongoing business processes automatically. It provides an easy-to-use yet powerful solution for identifying out-of-sync data before it negatively impacts the business.

With its advanced data comparison and validation capabilities, Oracle GoldenGate Veridata empowers businesses to confidently make critical decisions based on reliable, up-to-date information. Whether it's mission-critical systems, data migrations, or disaster recovery, Veridata ensures data integrity, reduces downtime, and enhances overall productivity.

1.2 Concepts

Get familiar with the following concepts and other commonly used terms before you get started with Oracle GoldenGate Veridata.

To begin using Oracle GoldenGate Veridata, you need to create some objects that identify the data that you want to compare and which help you to manage your work. Create these objects in the following order:

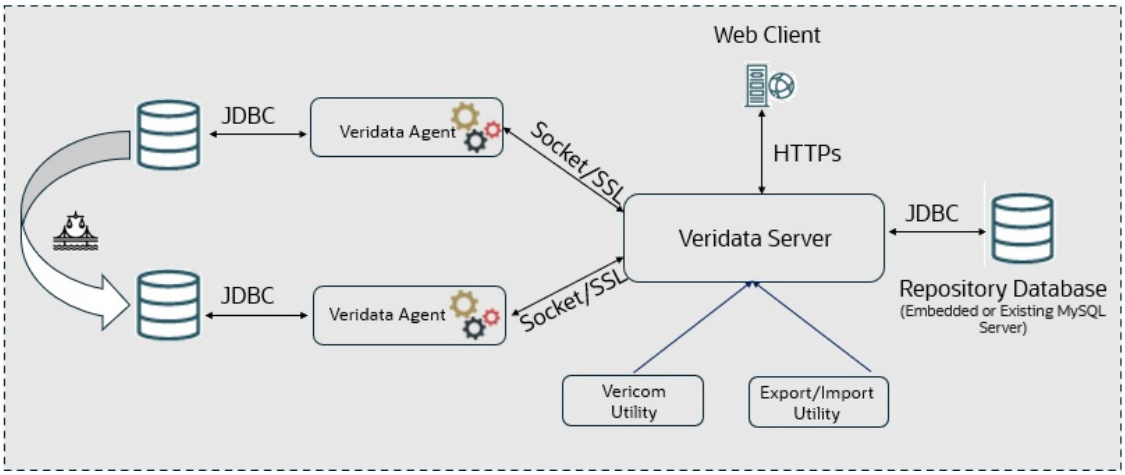
1. **Connections**: Define a Connection to the source and target databases that contain the data you want to compare. The Oracle GoldenGate Veridata server uses the Connection information to communicate with the Oracle GoldenGate Veridata agent.
2. **Groups**: Groups are logical containers for one or more Compare Pairs. A Group is associated with a set of connections to the source and target data.
3. **Compare Pairs**: A Compare Pair is the logical relationship between a source and target table for the purpose of comparing their data. Compare Pairs are linked to Groups.
4. **Jobs and Repairs**: To run Compare Pairs, you must create a Job. The Job configuration determines which compare groups are processed. In the UI, you can repair jobs and also can generate SQL files for a few supported databases for out-of-sync jobs.
5. **Users, User Groups, and Roles**: You can manage users and user groups from the **User Management** page. You can create, delete, edit, users and user group. Users and user groups are assigned with various roles, which are assigned with various privileges. The following are the various roles assigned in Oracle GoldenGate Veridata: Administrator, Super User, Monitoring Operator, Detail Monitoring Operator, Repair Operator, Job Operator, and Command Line Operator.

6. **Delta Comparison**: In Oracle GoldenGate Veridata, the source and target tables are configured using compare pairs, which are grouped and added to a job to run the comparison. During the subsequent runs of a comparison job, the comparison of the tables can be performed based on what has changed in the tables from the previous job run; these jobs are Delta Processing Jobs.
7. **Profile**: A profile is a set of global processing parameters, each containing unique settings for a specific purpose. Oracle GoldenGate Veridata provides a default profile, but you can create your own profiles. You can create as many profiles as needed and associate them with any job or compare pair (to override the job profile). You can override profile assignments at run time.

1.3 Oracle GoldenGate Veridata Architecture

The Oracle GoldenGate Veridata architecture comprises the following:

Figure 1-1 Oracle GoldenGate Veridata Architecture



- **Veridata Agent**
  It is a light-weight Java program that runs either on DB systems or remotely. The Veridata agent interacts with the source or target DBs to fetch and return blocks of data rows, the column-level detail, and execute updates for repairs.
- **Veridata Server**
  Compares data, confirms out-of-sync data, repairs data and produces reports. It also coordinates the jobs across all other components.
- **Repository Database**
  Oracle GoldenGate Veridata 23c supports only MySQL Server database to store the metadata.

The architecture also consists of three main utilities.

- **Vericom Utility**
  It is a command line interface. Veridata Command Line Interface. All the administration tasks performed in GUI can be done via this utility.
- **Export and Import Utilities**
  Used to Export Configuration files of existing Oracle GoldenGate Veridata and import it to the Veridata 23c. The utilities are mainly used in the migration process of Oracele

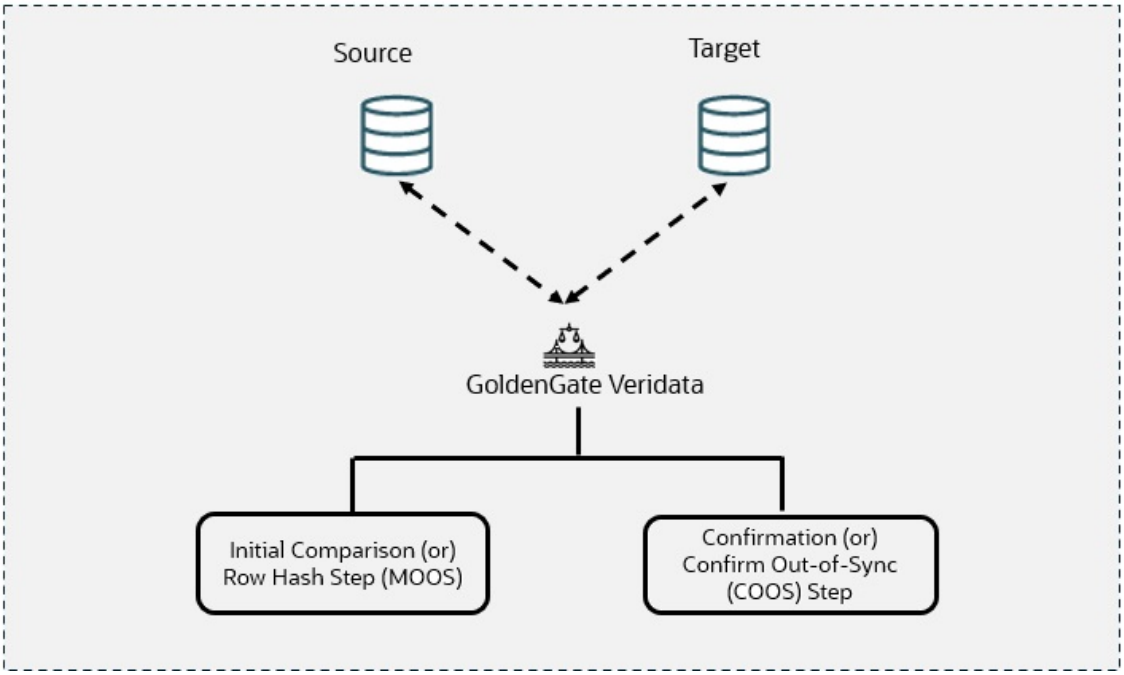
GoldenGate Veridata Servers. The Import Utility also helps in creating Compare Pairs, Groups, and Jobs using the Replicat Process parameter file.

1.4 How Oracle GoldenGate Veridata Works?

To compare data accurately while transactional and replication operations are taking place, Oracle GoldenGate Veridata uses a two-step process to maintain data accuracy:

1. **Initial step or Row hash step MOOS – Maybe Out-of-Sync**
2. **COOS - Confirm Out-of-Sync**

Figure 1-2 How Veridata Works?



- Initial Step or Row Hash Step in Maybe Out of Sync
- Confirmed Out of Sync

1.4.1 Initial Step or Row Hash Step in Maybe Out of Sync

The steps involved in Maybe Out of Sync (MOOS) are:

- Rows retrieved from source/target databases
- Data converted into standard format
- Hashes data and sent to Server
- Out-of-sync rows stored in MOOS queue

This marks the initial stage of the comparison process.

After the comparison is initiated, the Oracle GoldenGate Veridata agent retrieves rows from both source and target tables using a specified query.

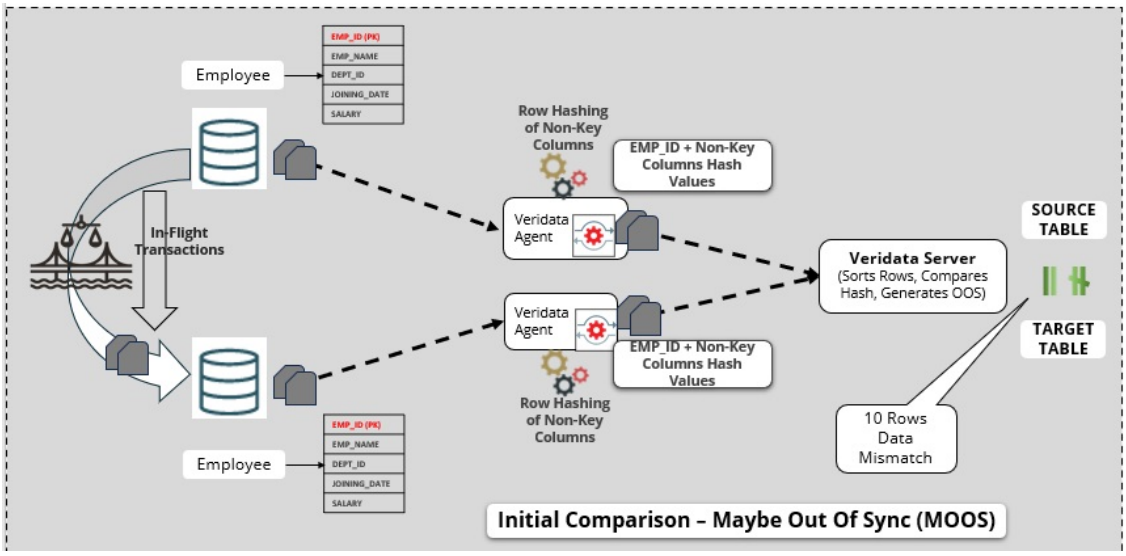
Oracle GoldenGate Veridata, being a heterogeneous data comparison and repair tool, ensures compatibility by standardizing data types if the source and target databases differ.

Data retrieval involves fetching primary key (PK) column values directly (actual values are fetched), while non-key columns are hashed to minimize network transfer for comparison. This unique hashing process streamlines the comparison process by providing a dependable and efficient means of determining similarities or differences between rows in the source and target databases.

Additionally, it is possible to alter the default hashing option. However, this adjustment may lead to decreased performance and increased network usage, particularly with a higher number of columns. After completing the initial comparison, if Oracle GoldenGate Veridata detects discrepancies in some rows, then it refrains from displaying the comparison results to the user. Instead, it stores these rows in the Maybe Out of Sync (MOOS) queue in memory. This precaution is necessary due to concurrent real-time replication, which may have some replication latency, potentially leading to the out-of-sync rows being in transit. Subsequently, replication will synchronize them again.

Example: Comparing data of the Employee table in both the source and target databases.

Figure 1-3 Initial step or Row hash step MOOS – Maybe Out-Of-Sync



The Employee table contains a primary key column `EMP_ID`.

1. Oracle GoldenGate Veridata retrieves the actual values of `EMP_ID` and hashes the values of other non-key columns.
2. The replication process is in progress. If there are any disparities, for example, if there are 10 rows being out of sync, then Oracle GoldenGate Veridata will not immediately confirm them as out of sync.
3. Given the real-time replication setup and concurrent data replication, it is possible that these 10 out-of-sync rows have yet to be replicated to the target. This might be due to latency.
4. The out-of-sync rows are stored in the MOOS queue in memory.

1.4.2 Confirmed Out of Sync

The steps involved in Confirmed Out of Sync (COOS) are:

- Ensures accurate results by comparing each row
- It happens in parallel with Row hash with configured latency threshold
- Confirm out of Sync requests can be batched
- Good performance for remote databases
- Controlled by a profile setting
- Executing the look ups in parallel

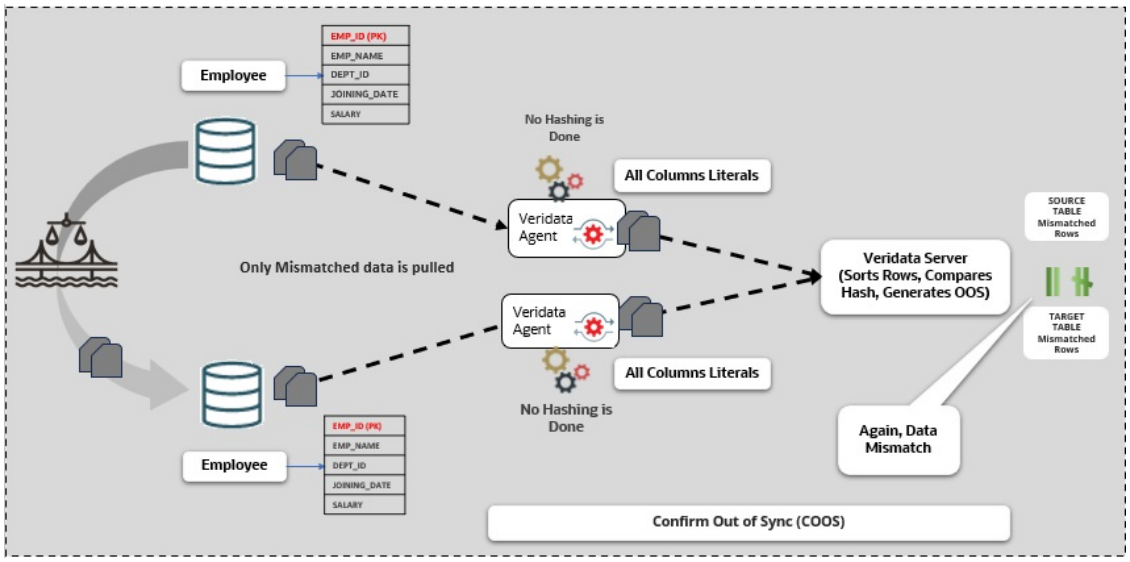
The verification process, known as confirmation or Confirmed Out of Sync (COOS), guarantees precise outcomes by validating the status of rows in a dynamic setting. This process entails making conditional inquiries on either the source or target using the rows obtained from the MOOS queue, with the status being assessed as one of the subsequent possibilities:

- **In Flight**: The row was initially not synchronized during the comparison process, but it has now been corrected. In this scenario, it is presumed that replication or a similar mechanism implemented the modification; however, Oracle GoldenGate Veridata could not verify the synchronization of the rows.
- **In Sync**: The values from the source row were transferred to the target row either through replication or a similar process. Even if the status shows as in-sync, it does not ensure that the rows are synchronized at any given moment, especially if the underlying tables are constantly changing. However, it does signify that replication is functioning properly.
- **Persistently out of sync**: After the initial comparison step occurred, the row remains unchanged, indicating it is likely out of sync.
  By default, confirmation processing takes place concurrently with the initial comparison step, but the confirmation of each row is delayed until after a specified replication latency threshold has passed. For example, if the latency is set to 60 seconds and an out-of-sync row is identified during the initial comparison step at 10:00, the confirmation step for that row is postponed until 10:01 to ensure replication can apply any pending changes. After the latency is considered, rows can be definitively marked as Out-of-Sync and cataloged in one or more out-of-sync reports.

Example: Comparing data of the Employee table in both the source and target databases.

After surpassing the latency threshold, the process moves to the second stage known as the COOS step. Here, only the rows that are out of sync are examined, without utilizing hashing. All values from both primary key and non-key columns are retrieved in their original form (actual values). The out-of-sync rows are compared value by value. If the same data inconsistencies persist after this stage, these rows are confirmed as out of sync. You are then presented with details of the out-of-sync rows for further action.

Figure 1-4 Confirmed Out of Sync (COOS)



> **Note:**
>
> The latency threshold is crucial in this context. In the previously mentioned scenario, it was established at 60 seconds, but it is adjustable manually. The **Delay Confirm-Out-Of-Sync** parameter within the Profile Configuration governs this threshold. Users are better acquainted with their replication environment and the daily latency they encounter. Hence, they can customize this parameter according to their requirements.

Delay Confirm-Out-Of-Sync By (seconds)

Delays the confirmation step by the specified number of seconds to account for replication lag. Delaying the confirmation step reduces the number of false out-of-sync results that occur because an updated source value was not replicated fast enough.

> **Note:**
>
> You have the option to bypass the two-step-comparison process altogether, specifically skipping the second step known as COOS.
> The parameter **Perform Confirm Out-Of-Sync Step** in the Profile Configuration allows users to decide whether to execute or bypass the second step when comparing data in Oracle GoldenGate Veridata.

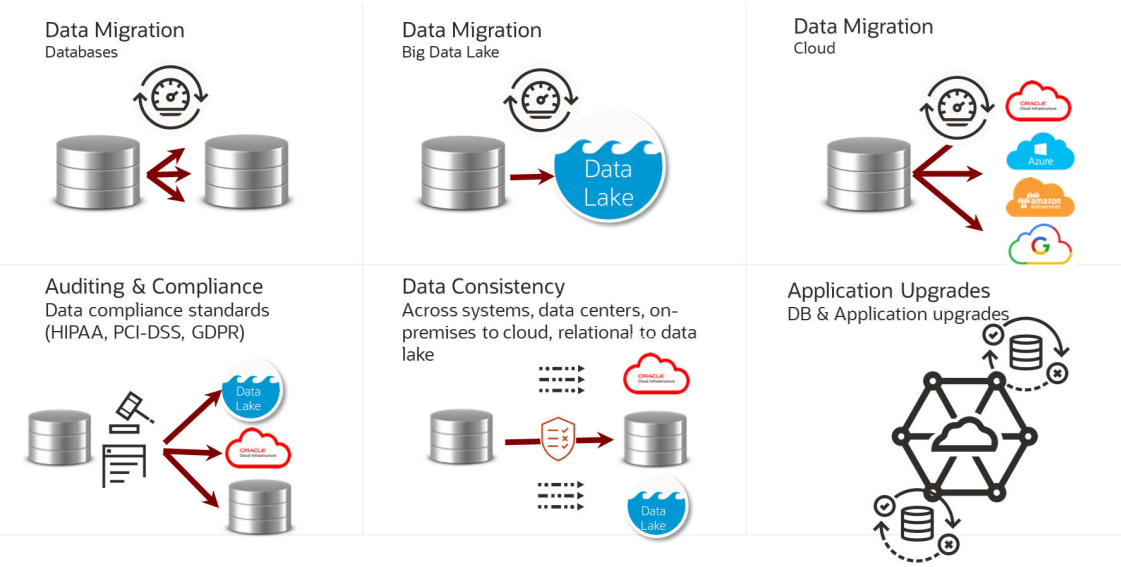
Perform Confirm-Out-of-Sync Step

Controls whether or not the confirmation step is performed. By default, it is performed. Clear the check box under Value to skip the confirmation step and only perform the initial (row hash) comparison. You might skip the confirmation step if, for example, activity on the source tables is quiesced or if replication is not continuously updating the target table(s).

1.5 Oracle GoldenGate Veridata Supported Use Cases

In today's complex, hybrid (Cloud and On-premises) IT environment with the same data stored in multiple locations or while migrating the data from one system or application to another, some data discrepancies are almost inevitable. If not discovered and addressed, bad data can lead to poor decision-making; failed service level agreements; and, ultimately, operational, financial, and legal risk.
Furthermore, for a comprehensive explanation, the following are the diverse use cases supported in Oracle GoldenGate Veridata:

Figure 1-5 Oracle GoldenGate Veridata Use Cases



- **Data Migration**: It is the process of transferring data from one location or one system to another. Data migration is a critical task in various scenarios, including:
  - **Database Upgrades**: When organizations upgrade their software or hardware systems, they often need to migrate data from the old system to the new one. For example, when transitioning to a new database management system (DBMS) or a newer version of an application.
  - **Cloud migration**: Many businesses are moving their data and applications to cloud platforms. Data migration in this context involves moving data from on-premises servers to cloud-based storage or from one cloud provider to another.
  - **Data center relocation**: When a company moves its data center to a new physical location, data migration is essential to ensure the continuity of operations.
    There are various factors involved in the Data Migration out of which Data Validation post migration is an important task. After the migration is complete, it is essential to verify that the data in the new system functions correctly and retains its integrity.
- **Auditing & Compliance**: As compliance standards evolve, corporations and their leaders are increasingly required to uphold exceptionally high levels of responsibility. Organizations must ensure their alignment with auditing and compliance standards and furnish data to regulatory authorities, such as Health Insurance Portability and Accountability Act (HIPAA), Payment Card Industry Data Security Standard (PCI-DSS), General Data Protection Regulation (GDPR), and others.

- **Data Consistency**: In any replication scenario, it is essential to ensure data consistency across all systems, whether it involves replicating data between data centers, migrating data from on-premises to the cloud, or transferring data from a relational database to a data lake. It is imperative that no data discrepancies occur.
- **Application Upgrades**: When conducting application upgrades, numerous Data Definition Language (DDL) and Data Manipulation Language (DML) operations are executed on the database tables. If this database has replication configured, it is imperative to ensure that all changes made in the source systems are seamlessly and accurately replicated to the target systems.

2
Prepare

2.1 Supported Databases for Compare

Oracle GoldenGate Veridata supports the following databases for comparisons:

- Oracle, ADW, ATP, and DBCS
- SQL Server
- MySQL, MariaDB (via MySQL JDBC connection string and driver)
- Teradata
- DB2 for i, DB2 LUW, and DB2 z/OS
- Non Stop (Enscribe and SQL/MP)
- Sybase Adaptive Server Enterprise (ASE)
- PostgreSQL

To find the latest information about Oracle GoldenGate Veridata release, including the list of certified database versions and operating systems, go to My Oracle Support at http://support.oracle.com.

2.2 Supported/Unsupported Databases for Repair

Oracle GoldenGate Veridata supports the following databases for repair functionality:

- Oracle, ADW, ATP, DBCS
- SQL Server
- Teradata
- DB2 for i, DB2 LUW, and DB2 z/OS
- Sybase Adaptive Server Enterprise (ASE)
- MySQL, MariaDB (via MySQL JDBC connection string and driver)
- Non Stop (Enscribe, SQL/MP)
- PostgreSQL

Oracle provides late-breaking updates, discussions, and best practices about pre-upgrade requirements, upgrade processes, post-upgrade, compatibility, repair, and interoperability. For information about the latest release of Oracle GoldenGate Veridata, including the list of certified database versions and operating systems, go to My Oracle Support at http://support.oracle.com.

2.3 Supported/Unsupported Databases for Download Repair SQL

Oracle GoldenGate Veridata supports the following target databases for Download Repair SQL functionality:

- Oracle
- SQL Server

2.4 Supported Datatypes

- Supported Datatypes - Oracle
- Supported Datatypes - Oracle 23ai
- Supported Datatypes - SQL Server
- Supported Datatypes - PostgreSQL
- Supported Datatypes - MySQL, MariaDB
- Supported Datatypes - DB2 LUW
- Supported Datatypes - DB2 for i
- Supported Datatypes - DB2 z/OS
- Supported Datatypes - Sybase
- Supported Datatypes - Teradata
- Generic Limitations and Clarifications

2.4.1 Supported Datatypes - Oracle

The Oracle GoldenGate Veridata supports the listed datatypes for the Oracle database:

- CHAR
- NCHAR
- VARCHAR2
- VARCHAR
- NVARCHAR2
- NUMBER
- DATE
- BLOB
- CLOB
- NCLOB

- LONG
- RAW
- LONG RAW
- ROWID
- BINARY_FLOAT
- BINARY_DOUBLE
- DATE
- TIMESTAMP
- TIMESTAMP WITH TIMEZONE
- TIMESTAMP WITH LOCAL TIME ZONE
- UROWID
- XMLTYPE
- INTERVAL YEAR [(year_precision)] TO MONTH
- INTERVAL DAY [(day_precision)] TO SECOND [(fractional_seconds_precision)]
- FLOAT
- UDTs

For Oracle Boolean Data Type

- **Supported mapping MySQL Data Type using veridata number compare format**: Boolean and Number.
- **Supported mapping MySQL Data Type using veridata string compare format with limitation**: Char andVarchar.
- **Limitation**: Only uppercase values `TRUE` and `FALSE` are treated as in-sync.
- **Not supported MySQL Data Type**: Binary and Varbinary

Limitations and Clarifications:

Both Object types and Collection types UDTs are supported

2.4.2 Supported Datatypes - Oracle 23ai

The Oracle GoldenGate Veridata supports the listed datatypes for the Oracle 23ai database:

- VECTOR
- BOOLEAN

2.4.3 Supported Datatypes - SQL Server

The Oracle GoldenGate Veridata supports the listed datatypes for the SQL Server database:

- BIGINT
- BIT
- INT
- SMALLINT
- TINYINT

- DECIMAL
- MONEY
- SMALLMONEY
- NUMERIC
- FLOAT
- REAL
- CHAR
- NCHAR
- VARCHAR
- NVARCHAR
- TEXT
- NTEXT
- BINARY
- VARBINARY
- IMAGE
- DATE
- DATETIME
- DATETIME2
- SMALLDATETIME
- DATTIMEOFFSET
- TIME
- GEOGRAPHY
- GEOMETRY
- UNIQUEIDENTIFIER
- XML

For more information about conversion of SQL Server data types to Oracle data types, see Data Type Conversion in *Database Gateway for SQL Server User's Guide*.

2.4.4 Supported Datatypes - PostgreSQL

The Oracle GoldenGate Veridata supports the listed datatypes for PostgreSQL:

- Bit(n)
- Bit Varying(n)
- Boolean
- Char
- citext
- Varchar(n)
- Time with/without timezone
- Date

- Interval
- Bigint
- Serial
- Smallserial
- Bigserial
- Numeric
- Decimal
- Money
- Real
- Double precision
- cidr
- inet
- macaddr
- macaddr8
- uuid
- text
- bytea (binary)
- xml
- smallint
- integer
- json
- jsonb

Non-Supported PostgreSQL Data Types

- arrays
- box
- circle
- composite types
- line
- lseq
- object identifiers
- OID
- pg_lsn
- pseduo types
- Point
- path
- polygon
- range types

- tsvector
- tsquery
- enum
- domain

For Postgres BIT Varying Data Type

- **Supported mapping Oracle/MySQL Data Type using veridata string compare format**: Char and Varchar
- **Not supported Oracle/MySQL Data Type**: Binary, Varbinary, and Number

Limitations of Support

- **Bit(n)/Bit Varying(n)**:
  - **Heterogeneous**: As the source side, Bit(n)/Bit Varying(n) can only be mapped with a character type of the non-PostgreSQL target database. This is because, the leading "0" of the source data gets truncated in the target db during repair, for example, the target is a number type.
  - **Homogeneous**: None.
- **Network datatype (inet,cidr,uuid)**:
  - **Heterogeneous**: All string datatypes like char/nchar/varchar in source side and network datatypes like inet/cidr/uuid in PostgreSQL target side can only considered as hash column.
  - **Homogeneous**: None.
- **MAC/MAC8 datatype**:
  - **Heterogeneous**: All string datatypes like char/nchar/varchar in source side and datatypes like mac/mac8 in PostgreSQL target side, while repair operation, it is successful but it always save in the one format`xx:xx:xx:xx:xx:xx` even though the allowed format is multiple, that is `xxx:xxx:xxx:xxx or xx-xx-xx-xx-xx-xx`. Therefore, it always shows the OOS after repair whenever there is a different format from source side.
  - **Homogeneous**: None.
- **JSONB datatype**:
  - **Heterogeneous**: All string datatypes like char/nchar/varchar in source side and jsonb in PostgreSQL target side, after repair the order of json key/values are not stored in original order. Therefore, it will be OOS for next compare pair run.
  - **Homogeneous**: None.
- **Timestamp with timezone**:
  - **Heterogeneous**: PostgreSQL is not storing the timezone value in the database. Whenever a timestamp with timezone value is inserted, PostgreSQL converts the timestamp into UTC and inserts into it. While retrieving, the actual timezone is not known. Comparing it with the DBs like oracle, sqlserver which stores complete timestamp along with timezone result in OOS all the time.
- **Time/Time with Timezone**:
  - **Homogeneous**: PostgreSQL- PostgreSQL Time/Time with Timezone datatype comparison and repair has inconsistencies when the time column is primary key. Some of the rows may not be picked for compare and repair.

- **Real**:
  - **Heterogeneous**: Oracle GoldenGate for Veridata cannot support the compare pair of FLOAT (Oracle) to REAL (PostgreSQL). Some data is always OOS after repair due to the internal representation of the data type in db. Substitute pairs are binary float (Oracle) to real (PostgreSQL) and float (Oracle) to numeric (PostgreSQL).
  - Real does not store the value as exact number in database. For example, 0.8 is stored as 0.800000011920929. On compare it always shows extra values, but it does not impact the repair functionality.
- **Interval**:
  - Heterogeneous: During repair Veridata may insert/update 00 to column.
    Oracle GoldenGate Veridata currently uses CAST (? AS INTERVAL) as part of its insert and update queries. This cast defaults to INTERVAL SECOND. Therefore, when values like '05', '57', or '98' are used as cast, they result in values, such as:
    * 5 --> 00:00:05
    * 57 --> 00:00:57
    * 98 --> 00:01:38

    Therefore, if these values are inserted/updated to a column with INTERVAL HOUR, the result is 00. The same is for other type of INTERVAL.
- **citext**
  Repair function is not fully supported when citext is mapped to a string-like data type on either source or target side. When citext is on source side,following change is needed for repair to work: go to **Profile Configuration > Edit Existing Profile > Repair**, and unselect the **Check Changed Values** setting.

2.4.5 Supported Datatypes - MySQL, MariaDB

The Oracle GoldenGate Veridata supports the listed datatypes for the MySQL, MariaDB databases:

- BIGINT
- BINARY
- BIT
- BLOB
- CHAR
- DATE
- DATETIME
- DECIMAL
- DOUBLE
- ENUM
- FLOAT
- INT
- INTEGER
- LONGBLOB
- LONGTEXT

- MEDIUMBLOB
- MEDIUMINT
- NUMERIC
- REAL
- SET
- SMALLINT
- TEXT
- TIME
- TIMESTAMP
- TINYBLOB
- TINYINT
- TINYTEXT
- VARCHAR
- VARBINARY
- YEAR

For MySQL BIT Data Type

- **Supported mapping Oracle/Postgres Data Type using veridata number compare format**: Number
- **Supported mapping Oracle Data Type using veridata string compare format with limitation**: Char and Varchar.
- **Limitation**: not applicable to compare value with leading 0, such as `0001101`
- **Supported mapping Postgres Type using veridata string compare format with limitation**: Char, Varchar and Bit varying
- **Limitation**: not applicable to compare value with leading 0, like `0001101`
- **Not supported Oracle Data Type**: Binary and Varbinary

Limitations and Clarifications

- For Oracle GoldenGate Veridata repair to properly throw errors when invalid or missing values are detected, user should set SQL Mode to "strict". This can be set on global level when MySQL starts, or session level via connection string in agent.properties, for example: `database.url=jdbc:mysql://host:3306?sessionVariables=sql_mode=(select concat(@@SESSION.sql_mode,',STRICT_TRANS_TABLES'))`
- When using `REAL` as primary key, MySQL client and Veridata may not able to retrieve certain rows (for example, `WHERE `PKCOL` = '-99.9999')`, Oracle GoldenGate Veridata skips these rows during compare and repair.
- When using nonstandard `FLOAT(M,D)` and `DOUBLE(M,D)`, the `(M,D)` in DDL need to match the actual precision and scale stored, and both Source and Target need to be MySQL.

2.4.6 Supported Datatypes - DB2 LUW

The Oracle GoldenGate Veridata supports the listed datatypes for the DB2 LUW database:

- BIGINT

- BLOB
- CHAR
- CHAR FOR BIT DATA
- CLOB
- DATE
- DBCLOB
- DECFLOAT
- DECIMAL
- DOUBLE
- FLOAT
- GRAPHIC
- INTEGER
- LONG VARCHAR
- LONG VARCHAR FOR BIT DATA
- LONG VARGRAPHIC
- NUMERIC
- REAL
- SMALLINT
- TIME
- TIMESTAMP
- VARCHAR
- VARCHAR FOR BIT DATA
- VARGRAPHIC
- XML

2.4.7 Supported Datatypes - DB2 for i

The Oracle GoldenGate Veridata supports the listed datatypes for DB2 for i:

- BIGINT
- BLOB
- CHAR
- CHAR FOR BIT DATA
- CLOB
- DATE
- DBCLOB
- DECFLOAT
- DECIMAL
- DOUBLE

- FLOAT
- GRAPHIC
- INTEGER
- LONG VARCHAR
- LONG VARCHAR FOR BIT DATA
- LONG VARGRAPHIC
- NUMERIC
- REAL
- ROWID
- SMALLINT
- TIME
- TIMESTAMP
- VARCHAR
- VARCHAR FOR BIT DATA
- VARGRAPHIC
- XML

Limitations and Clarifications

- The decimal part of REAL value from the wldb2 jdbc driver always store in the 16 digits precision, for example, 0.8 converts `0.800000011920929` and compare will always fail if the decimal values are present.

2.4.8 Supported Datatypes - DB2 z/OS

The Oracle GoldenGate Veridata supports the listed datatypes for the DB2 z/OS database:

- BIGINT
- BINARY
- BLOB
- CHAR
- CHAR FOR BIT DATA
- CLOB
- DATE
- DBCLOB
- DECFLOAT
- DECIMAL
- DOUBLE
- FLOAT
- GRAPHIC
- INTEGER
- LONG VARCHAR

- LONG VARCHAR FOR BIT DATA
- LONG VARGRAPHIC
- NUMERIC
- REAL
- ROWID
- SMALLINT
- TIME
- TIMESTAMP
- TIMESTAMP WITH TIME ZONE
- VARBINARY
- VARCHAR
- VARCHAR FOR BIT DATA
- VARGRAPHIC
- XML

2.4.9 Supported Datatypes - Sybase

The Oracle GoldenGate Veridata supports the listed datatypes for the Sybase database:

- bigdatetime
- bigint
- bigtime
- binary
- bit
- char
- date
- datetime
- decimal
- float
- image
- int
- money
- numeric
- real
- smalldatetime
- smallint
- smallmoney
- sysname
- text

- time
- timestamp
- tinyint
- unichar
- unitext
- univarchar
- unsigned bigint
- unsigned int
- unsigned smallint
- varbinary
- varchar

2.4.10 Supported Datatypes - Teradata

The Oracle GoldenGate Veridata supports the listed datatypes for the Teradata database:

- ARRAY
- BIGINT
- BLOB
- BYTE
- BYTEINT
- CHAR
- CLOB
- DATE
- DECIMAL
- DISTINCT
- FLOAT
- GRAPHIC
- INTEGER
- INTERVAL DAY
- INTERVAL DAY TO HOUR
- INTERVAL DAY TO MINUTE
- INTERVAL DAY TO SECOND
- INTERVAL HOUR
- INTERVAL HOUR TO MINUTE
- INTERVAL HOUR TO SECOND
- INTERVAL MINUTE
- INTERVAL MINUTE TO SECOND
- INTERVAL MONTH

- INTERVAL SECOND
- INTERVAL YEAR
- INTERVAL YEAR TO MONTH
- LONG VARCHAR
- LONG VARGRAPHIC
- NUMBER
- SMALLINT
- STRUCT
- TIME
- TIME WITH TIME ZONE
- TIMESTAMP
- TIMESTAMP WITH TIME ZONE
- VARBYTE
- VARCHAR
- VARGRAPHIC

2.4.11 Generic Limitations and Clarifications

This topic lists a few generic limitations and clarifications:

- As the floating-point data types are approximate values by definition, Oracle GoldenGate Veridata UI may display slightly different values than expected (for example, extra decimal values)
- For database with TIME data type that supports value outside range of 00:00:00 to 23:59.59, the compare will only work when Source and Target are the same database.
- Whenever there is a precision for any datatype, such as REAL/FLOAT or Double on the database side, Oracle GoldenGate Veridata rounds off the lower precision present either on the source or the target database. Veridata considers the lower precision of data during comparison phase.
- Oracle GoldenGate Veridata internally converts decimal numbers with a large scale by truncating tail digits. The limitations applies to all databases except Oracle.

  This is because Oracle GoldenGate Veridata utilizes a template to mask decimal numbers, which, in cases of a large scale, leaves few digits for representing the scale part. For example, a number like `6.710123456789e-26`, with a scale of 38 digits, would be truncated by Oracle GoldenGate Veridata to retain the first 15 digits, converting it to 0. For scenarios where preserving all digits of the scale is crucial, Oracle recommends using the string or other compare formats instead of the number compare format when configuring the compare pair.

2.5 Oracle GoldenGate Veridata Agent System Requirements

One Oracle GoldenGate Veridata Agent must be installed for each database instance that contains data that is to be compared. At minimum, therefore, you will install two agents — one to retrieve source rows and one to retrieve target rows (unless you are comparing data within the same database instance). One agent can retrieve rows from multiple databases or

schemas within a given database instance. However, one agent cannot retrieve rows from different database instances.

Comparing Multi-Byte Data

The following considerations apply when you are comparing tables with multibyte data:

- A Java agent should be used for all platforms except NonStop, which has only a C-agent.
- The Java agent uses the UTF-8 character for comparing character data. Out-of-sync data is written to the report file using the UTF-8 character set.

This topic contains the following:

- Disk and Memory Requirements for the Agent Component
- Database Privileges for the Agent Component

2.5.1 Disk and Memory Requirements for the Agent Component

- The agent requires at least 1GB of RAM.
- The disk space requirements for the Oracle GoldenGate Veridata Agent vary by platform, but up to 200 MB may be required. On UNIX and Linux, additional space might be required to install the Java environment (if not already installed).
- The main consumers of processing resources are the row sorting operations that are required during a comparison. To improve performance, you might need to increase the temporary memory space in the database if the columns that are being used as keys are not a native unique index or primary key. You specify the columns to use as keys when configuring Oracle GoldenGate Veridata.
- Using server-side sorting instead of database sorting might reduce the load on the database server and improve comparison performance, depending on the number of rows, the indexes defined, the keys used, and the way the database is tuned. See Disk and Memory Requirements for the Server Component

2.5.2 Database Privileges for the Agent Component

Oracle GoldenGate Veridata Agent makes use of a database login, which must be created before you can run comparisons. You provide the login and password when you configure connection objects in the Oracle GoldenGate Veridata Web interface. The following are the database privileges that are required for the database user.

Required database privileges for Oracle GoldenGate Veridata Agent

DB2

- `SELECT` privileges on the tables that will be compared.

Oracle

- `GRANT CONNECT`
- `GRANT SELECT` on the tables to be compared. It is recommended, but not necessary, to `GRANT SELECT ANY TABLE`.
- `SELECT_CATALOG_ROLE`
- `EXECUTE_CATALOG_ROLE` (for `GET_TAG` and `SET_TAG` procedures)

- `CREATE TABLE` (for COOS Join in case of tables with no primary/unique constraint)
- `CREATE TABLESPACE` (for COOS Join in case of tables with no primary/unique constraint)
- `GRANT UPDATE`, `DELETE`, `INSERT` (for COOS Join in case of tables with no primary/unique constraint)

NonStop SQL/MP and Enscribe

- Read access to the SQL/MP system catalog (for queries to `CATALOGS` table).
- Read access to the SQL/MP catalogs that you want Oracle GoldenGate Veridata to use.
- Read access to the DDL dictionaries that you want Oracle GoldenGate Veridata to use.
- Read access to the Enscribe and SQL/MP tables that will be compared.
- Read, write, create, purge permissions for the Oracle GoldenGate Veridata report and trace files, and access to the sub volumes where they are installed.

SQL Server

- `db_datareader` or the equivalent on the tables to be compared.
- `VIEW DEFINITION` in the databases to be compared.
- The database must allow SQL Server authentication.

Sybase ASE

- Access to the databases to be compared.
- `SELECT` privileges on the tables to be compared.
- `SELECT` privileges on the `sysdatabases` system table in the master database to view the list of databases available in the server.

Teradata

- `SELECT` privileges on the tables to be compared.

BigData Hive

SELECT privileges on the tables to be compared.

Required Database Privileges for Using the Repair Feature

For all databases, the database user must have the `UPDATE`, `INSERT`, and `DELETE` privileges on the tables to be repaired.

For Sybase database, if the table has triggers and suppression of triggers enabled, the database user must have the `replication_role` privilege.

For SQL Server database, If the table being repaired has 'identity columns', the Repair User specified must be either the table owner, or should have `ALTER` permission on the table that is being repaired.

2.6 Oracle GoldenGate Veridata Server System Requirements

This section describes the installation location, additional programs, disk, memory, and repository requirements for Oracle GoldenGate Veridata Server.

Oracle GoldenGate Veridata server is designed to run on a dedicated Linux host only. Ensure to stop all other applications and remove any MySQL instances before server install to avoid port conflicts.

Oracle recommends that you use the certification matrix and system requirements documents with each other to verify that your environment meets the requirements for installation. See GoldenGate Certifications.

- Location for the Server Component
- Disk and Memory Requirements for the Server Component

2.6.1 Location for the Server Component

The server and web user interface components are installed from one installation program on Linux systems. The installer includes all files that are needed to run those programs. One installation can be used for comparisons among all of the supported databases, but multiple installations can be used as needed.

Do not install the server and web user interface components on a NonStop system. To use Oracle GoldenGate Veridata for NonStop databases:

- Install the server and web user interface components on a supported Linux system.
- Make certain that this system has access over high-speed network connections to the NonStop systems.

2.6.2 Disk and Memory Requirements for the Server Component

The server component uses about 200 MB of fixed virtual memory for basic tasks. The remaining virtual memory is used for comparisons. The main consumers of processing resources on the Oracle GoldenGate Veridata machine are the row sorting operations of the initial comparison step when using server-side sorting.

Enough combined disk space and virtual memory is needed to store all of the rows that are sent for comparison from the source and target systems. To estimate the amount of memory per row:

```
((number of cols in key + 1) * 4) + 16 + (comparison width of a key col)
```

Where:

`comparison width of a key col` depends on the comparison format that is selected by Oracle GoldenGate Veridata (or a user override) to use for a comparison.

Comparison format data sizes:

Comparison Format	Data Size
Numbers	One byte for each significant digit. Leading zeros and trailing zeros after the decimal point (such as the right most zeros in 1234.500) are not counted.
Timestamp	19 to 32 bytes depending on the fractional precision.
Date	10 bytes.
Time	8 to 18 bytes depending on the fractional precision.
String	1 to 4 bytes per character for the UTF-8 encoding of the Java agent. The NonStop agent uses the database native character set.
Binary	The bytes as stored in the database.

For example, the number 109998877, if compared as a decimal float, would require:

`((1 + 1) * 4) + 16 + 9 = 33` bytes of memory for this row

> **Note:**
>
> This assumes that all non-key columns are compared by using a hash, not literally. More space is needed for literal comparisons

Oracle GoldenGate Veridata uses an external merge sort to sort the data. As data is received from the agent, the rows are sorted in memory. When a memory buffer is full, the sorted rows are written to disk.

In order to sort the data, the sort process matches the initial data set size for temporary storage space. The required amount of temporary space is determined by the number of rows, the row size, and the amount of available sort memory. The following cases illustrate the different modes of the sort depending on the available resources.

- **In-Memory Sort:** This sorts the data entirely in memory and is the fastest method, but the memory requirements may exceed what is available. The sort memory must be approximately 2.5 times larger than the size of the data set.
- **One Disk Pass:** This sorts data and writes to the disk only once. It requires sort disk space equal to the size of the data set. This process is almost as fast as the in-memory sort and the memory requirements are lower. In general the Oracle GoldenGate Veridata server can write the rows to disk faster than the agent can read them from the database.
- **Two Disk Passes:** This sorts and writes to the disk twice, requiring sort disk space twice the size of the data set. Although the disk requirement is greater, very large data sets can be sorted with a reasonable amount of memory.
- **Three or More Disk Passes:** After all of the rows have been received from the agent, additional sorting may be required before the rows are ready for the final write to disk. If it is necessary to access the disk three or more times, the required sort space will be three or more times the data set size. This is slow and should be avoided.

Beyond this allocation, memory is required for storing rows during the second step of processing, the confirmation step. This can be up to 20 MB if you expect a large number of rows to require confirmation, as is usually the case when replication latency is very high. These rows are staged in the main memory before they are confirmed.

On 64-bit systems, more memory can be addressed, so more data can be stored in main memory instead of on slower disk devices. The memory that is used in the initial comparison step is not necessarily all released at once to be available for the confirmation step. Consequently, some memory will be shared between processes. When the sort cannot hold all of the rows in memory, it uses disk storage.

When deciding how much memory to allocate, be aware of the following ways that you can manage it with parameter settings within the Oracle GoldenGate Veridata application:

- The temporary space should be located on a reasonably fast file system. A network file system located on a remote server may slow the comparison processing.
- You can increase disk I/O performance by specifying multiple temporary directories with profile settings. For maximum benefit, put the directories on different physical disks.
- You can use a profile setting to terminate the confirmation step after a given number of out-of-sync rows, to work around resource limitations.

- Additional memory properties can be controlled with server parameters. See Server Parameters.

2.7 Oracle GoldenGate Veridata Distribution

This topic details how to download the files of Oracle GoldenGate Veridata files and C-Agent.

- Downloading Oracle GoldenGate Veridata
- Downloading Oracle GoldenGate Veridata C-Agent and Java Agent

2.7.1 Downloading Oracle GoldenGate Veridata

A distribution is an archive file containing an installer; when you run the installer, the set of Oracle GoldenGate Veridata components and feature that are included with the distribution are installed. You will need a certified JDK on your system in order to be able to run the installer. For more information, see **Prerequistes** in Installing and Running Oracle GoldenGate Veridata

You can download Oracle GoldenGate Veridata from the Oracle GoldenGate Downloads page at https://www.oracle.com/middleware/technologies/goldengate-downloads.html and from the Oracle Software Delivery Cloud site, at https://edelivery.oracle.com/osdc/faces/SoftwareDelivery.

The following table describes the products and feature sets in Oracle GoldenGate Veridata.

Table 2-1 Oracle GoldenGate Veridata Product and Feature Sets

Product	Feature Set	Description
Oracle GoldenGate Veridata	Veridata Server	The Veridata Server component includes the web server, the web application, and command-line utilities such as `vericom`.
	Veridata Agent	This component installs only the Veridata Agent.
Internal Features	OPatch	The OPatch utility is a tool that allows the application and rollback of interim patches to Oracle products.

2.7.2 Downloading Oracle GoldenGate Veridata C-Agent and Java Agent

Oracle GoldenGate Veridata C-Agent and the Java Agent platforms are available for download on Oracle Technology Network Oracle GoldenGate Downloads page at:

`http://www.oracle.com/technetwork/middleware/goldengate/downloads/index.html`

3
Install

3.1 Installing and Running Oracle GoldenGate Veridata

This article describes all the steps to install, configure, and uninstall Oracle GoldenGate Veridata.

Prerequisites

- If your system does not have graphical desktop, then ensure that you setup VNC and connect to it using a VNC client for graphical desktop: `vncserver`
- Ensure to install the JDK software.
  1. Go to http://www.oracle.com/technetwork/java/javase/downloads/index.html.
  2. Select JDK version 17 and click **Download**. You can change the default installation location.
  3. Set `JAVA_HOME`.
     Korn and bash shells:

     ```
     export JAVA_HOME=jdk-install-dir
     export PATH=$JAVA_HOME/bin:$PATH
     ```

     Bourne shell:

     ```
     JAVA_HOME=jdk-install-dir
     export JAVA_HOME
     PATH=$JAVA_HOME/bin:$PATH
     export PATH
     ```

     C shell:

     ```
     setenv JAVA_HOME jdk-install-dir
     setenv PATH $JAVA_HOME/bin:$PATH
     ```
- Ensure not to run additional application on the same host as it may have a strong impact on the performance of Oracle GoldenGate Veridata. Note that Veridata is a Java application designed to run on dedicated host with optimal heap and JVM settings.

Installing Oracle GoldenGate Veridata

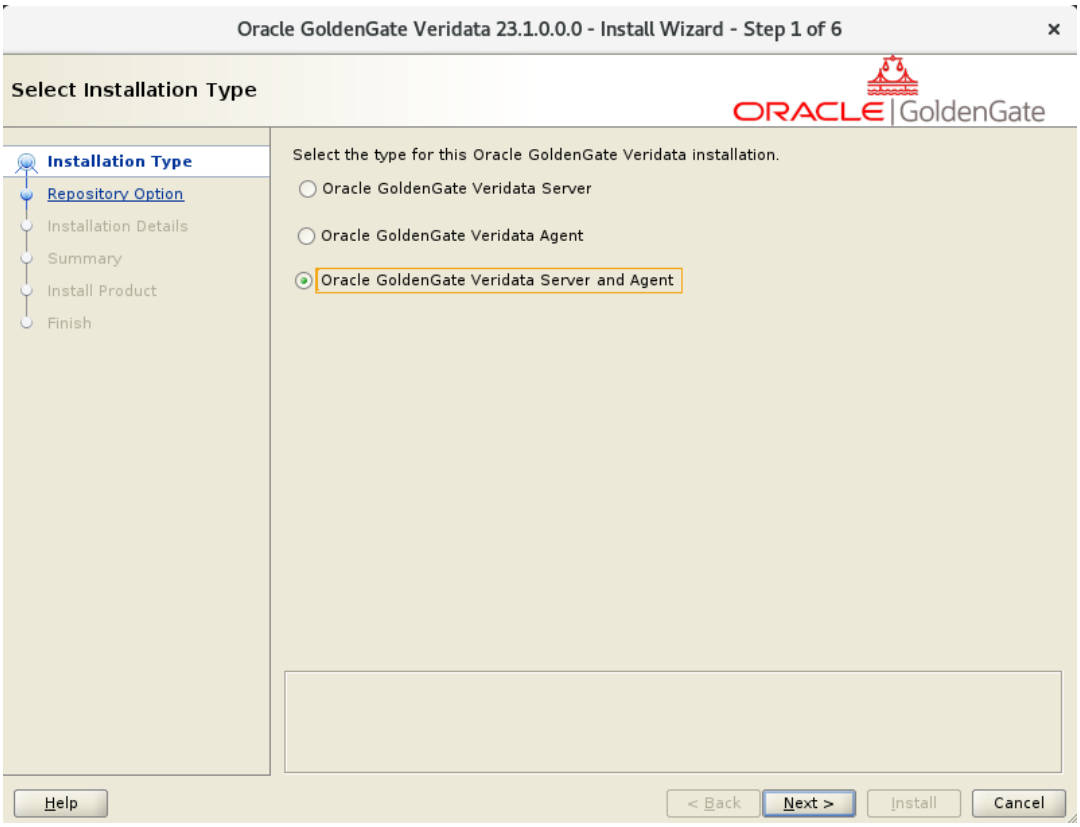
To install Oracle GoldenGate Veridata

1. Download the installer. See Downloading Oracle GoldenGate Veridata.
2. Run the following command to launch the installer UI:

```
unzip fbo_oggvdt_linux_services_shiphome.zip
cd fbo_oggvdt_linux_services_shiphome/Disk1/
./runInstaller
```

3. In the **Select Installation Type** page, select one of the following 3 options to set the type for the Oracle GoldenGate Veridata installation and click **Next**:
   - Oracle GoldenGate Veridata Server
   - Oracle GoldenGate Veridata Agent
   - Oracle GoldenGate Veridata Server and Agent

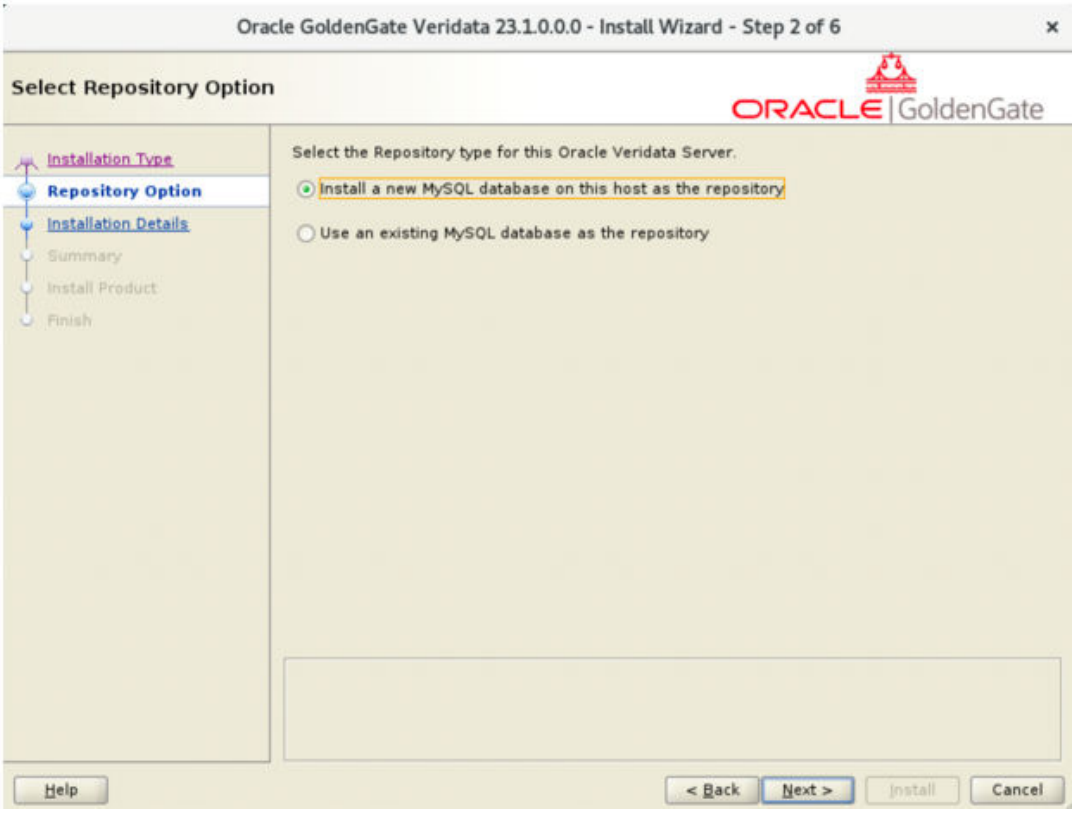
Figure 3-1 Select the Installation Type



4. If the Installation Type is chosen as **Oracle GoldenGate Veridata Server** or **Oracle GoldenGate Veridata Server and Agent** in step 3, then the **Repository Option** page is displayed. Select one of the following 2 options to set the Repository type for the Oracle GoldenGate Veridata Server and click **Next**:
   - Install a new MySQL database on this host as the repository
   - Using an existing MySQL database as the repository

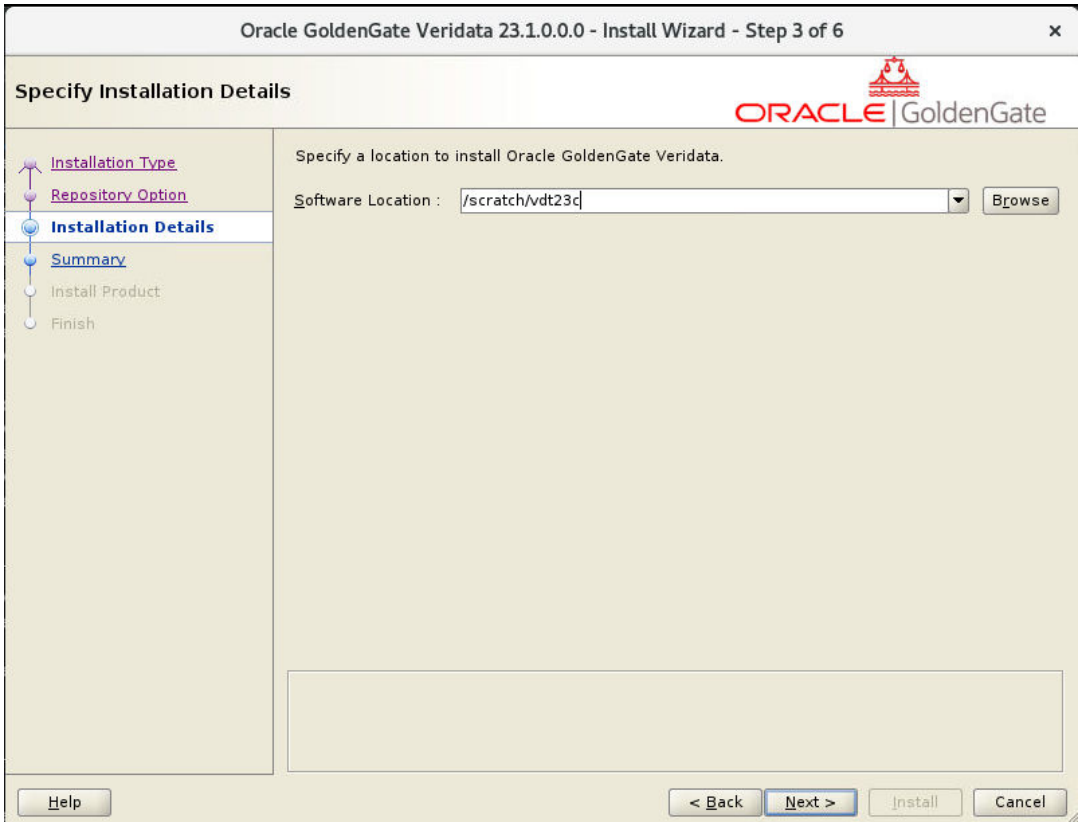
> **Note:**
>
> If the **Using an existing MySQL database as the repository** option is selected, then ensure that the existing MySQL database is a commercial version 8.0.34 and above. It can be a user installed or system preinstalled MySQL database. However, ensure that it is not an embedded MySQL database from a prior Oracle Veridata installation.

Figure 3-2 Select Repository Option



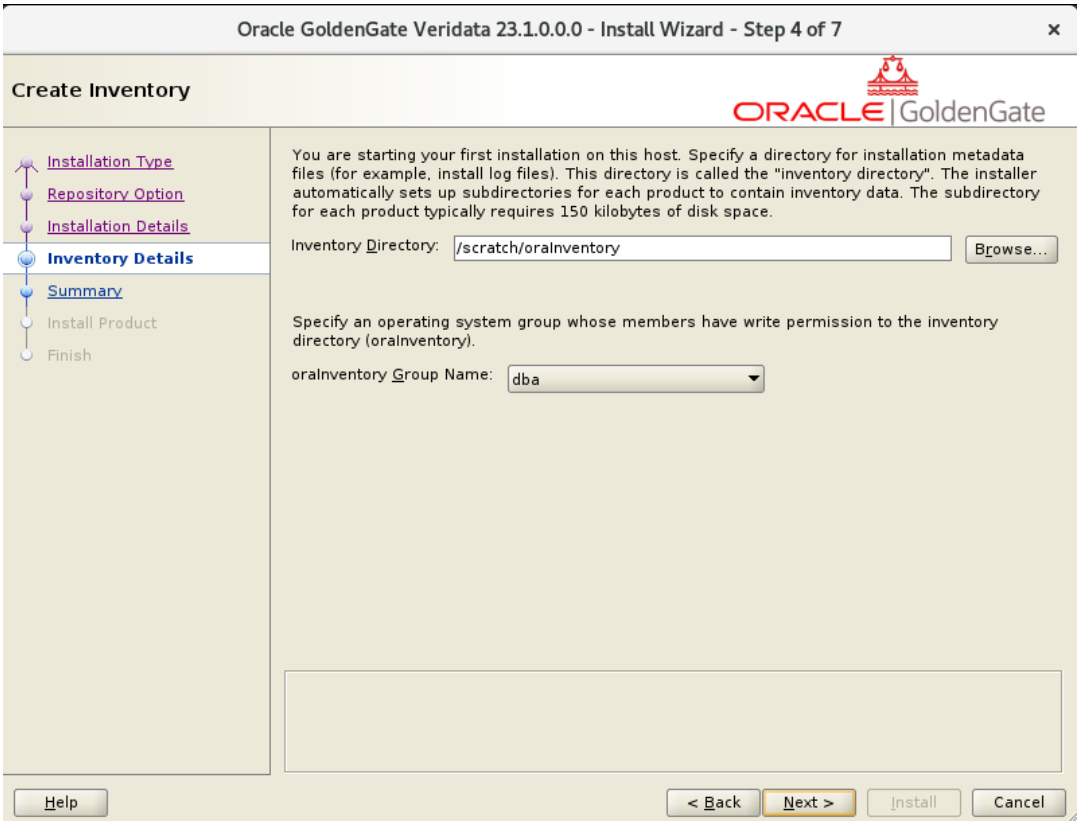
5. In the **Specify Installation Details** page, specify the **Software Location** to install Oracle GoldenGate Veridata. For example, **/scratch/vdt23c**

Figure 3-3 Specify Installation Details



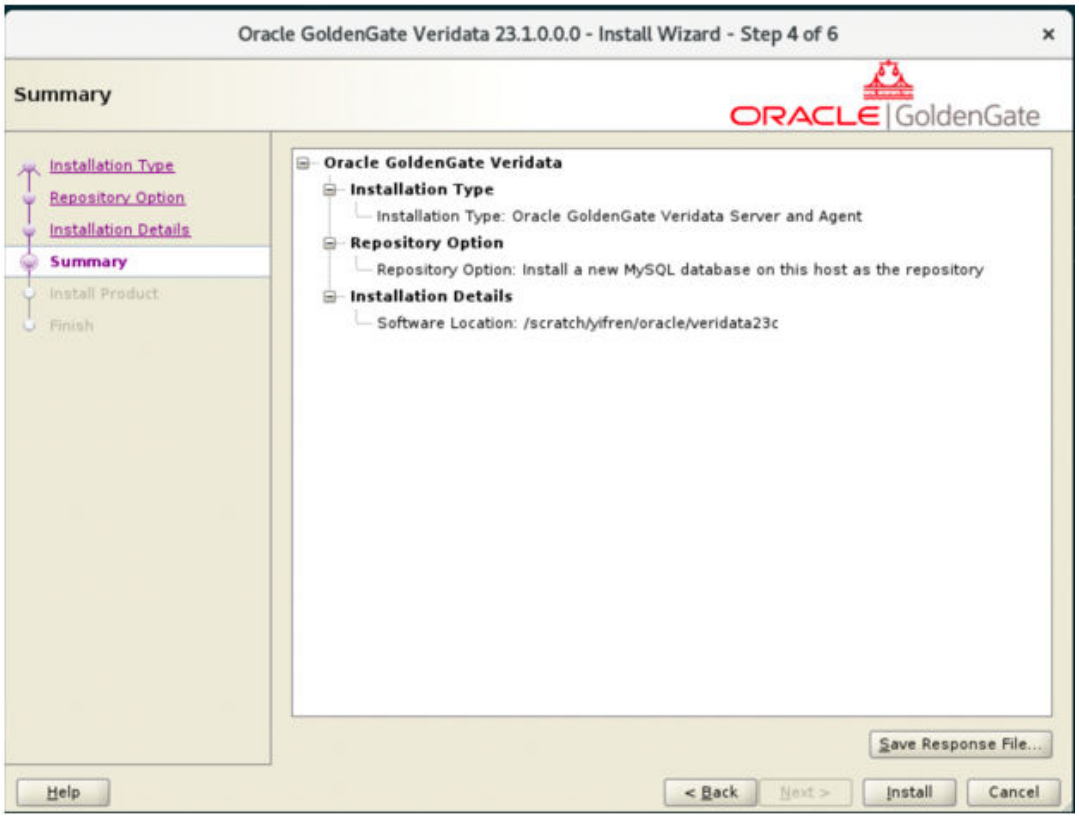
6. In the **Create Inventory** page, you can either retain the defaults or provide or enter a new directory for the inventory and click **Next**.

Figure 3-4 Inventory Details



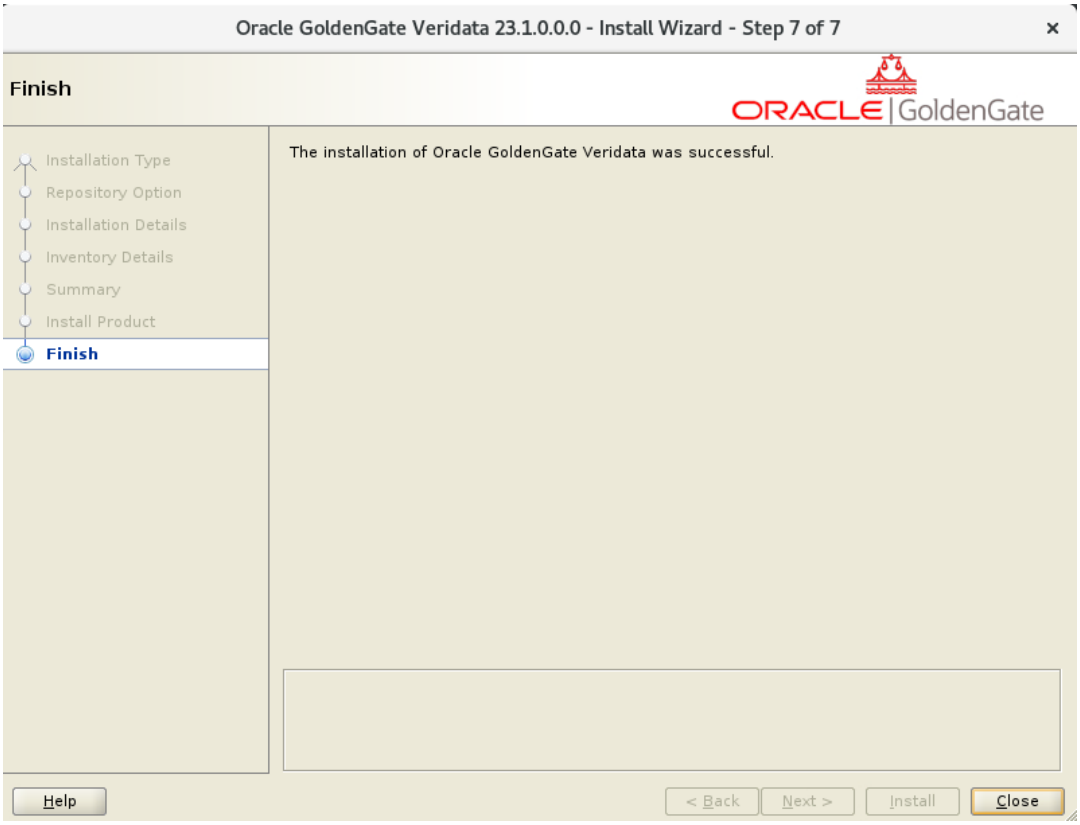
7. Review the content in the **Summary** page, click **Install**.

Figure 3-5 Installation Summary



8. If you want to save the response file, then click **Save Response File** before you begin with installation.
   At the end of the installation process, if the Oracle GoldenGate Veridata installation is complete, then the Installation Complete page is shown as follows:

Figure 3-6 Installation complete



- Logs

3.1.1 Logs

Server Installation Logs

The Server installation logs are created in *VERIDATA_HOME*/cfgtoollogs/ directory, where *VERIDATA_HOME* is the home directory where Oracle GoldenGate Veridata is installed..

Server Logs

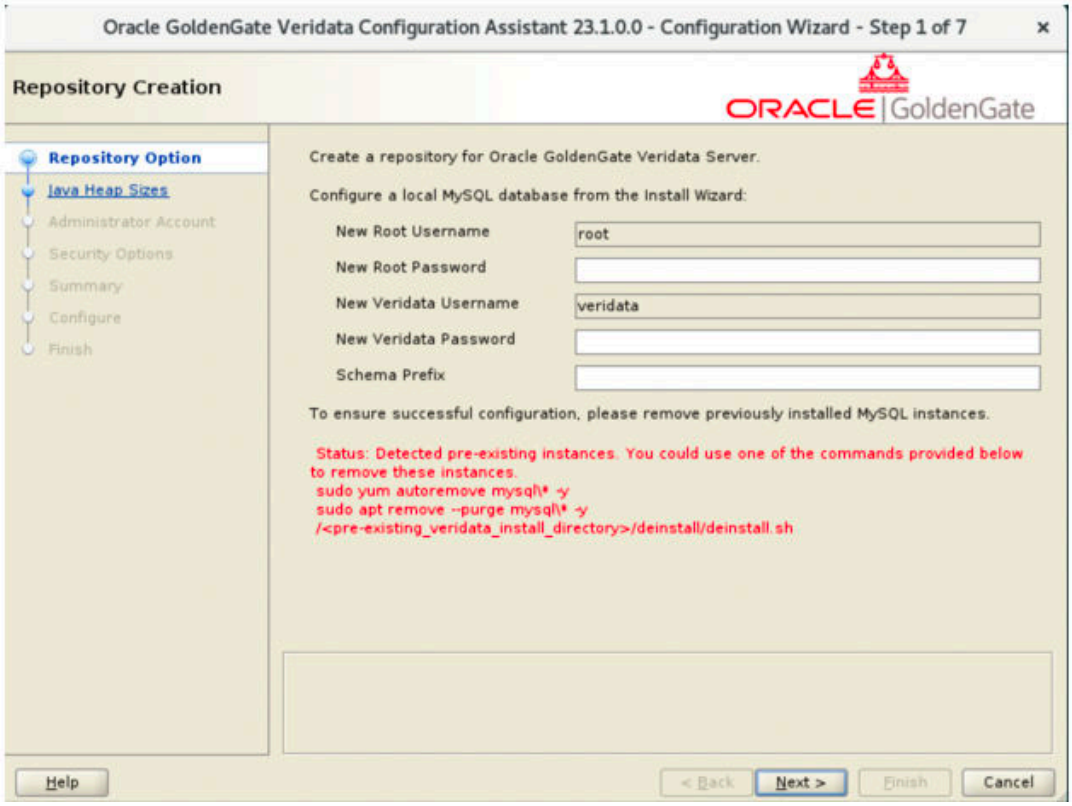
The Server log files are created in this directory: *VERIDATA_HOME*/veridata/logs. The logs directory contains files of format, such as vdtapi.log.0, vdtapi.log.1, or vdtapi.log.2, and other related logs. Note that vdtapi.log.0 will always contain the latest logs.

3.2 Running the Configuration Assistant

1. After the Oracle GoldenGate Veridata installation is complete, navigate to the bin directory, and run vdtca.sh to display the Configuration Assistant.
2. If MySQL is a new install from Installer, then in the **Repository Creation** page, enter the **New Root Password**, and **New Veridata Password**. Ensure to pick a unique **Schema Prefix** to allow multiple Veridata installation to point to the same MySQL Repository. To ensure successful configuration, remove previously-installed MySQL instances.

> **Note:**
>
> If the Configuration Assistant detects a pre-existing MySQL instance in the system, it will display the information with red text shown as follows:

Figure 3-7 Repository Creation Page for a New MySQL Install



> **Note:**
>
> If the Configuration Assistant shows a warning message saying it has detected a pre-existing MySQL for the new MySQL installation option, then run the following command: `yum list installed | grep mysql` and verify if any MySQL database or package is installed in the system. If you are sure that the MySQL database does not exist in the system, then you can safely ignore the warning and proceed with the configuration. For example, `pcp-pmda-mysql.x86_64` package can cause a false warning.

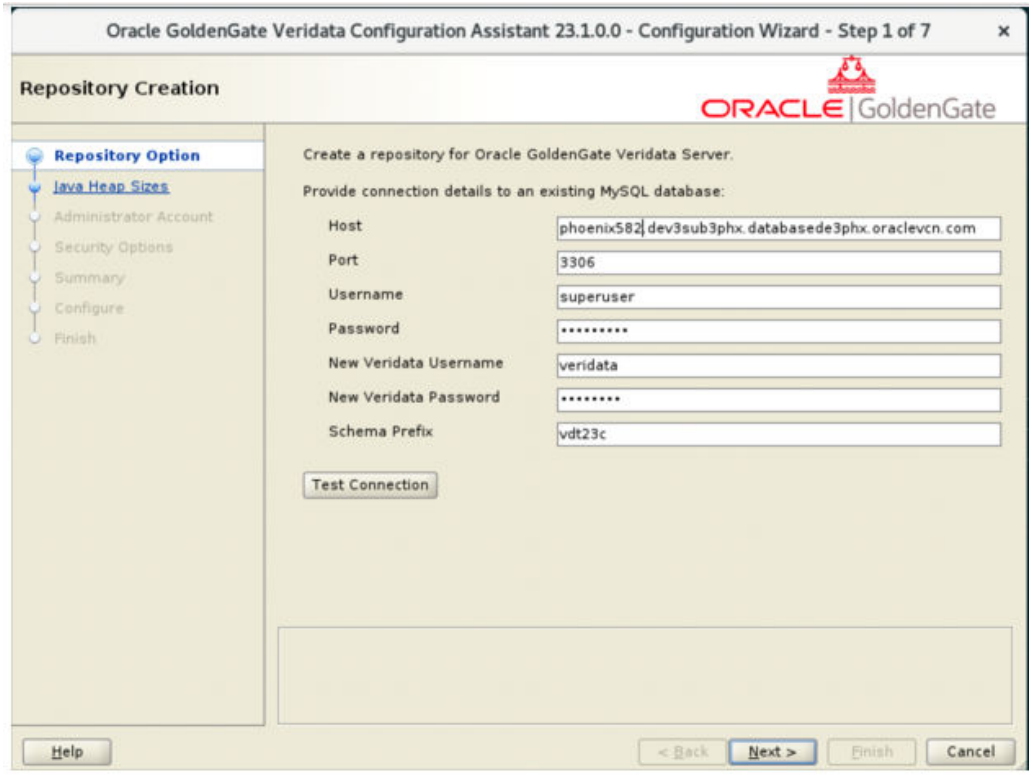
If MySQL is an existing database, then in the **Repository Creation** page, enter the database details for all the text fields.

- The New Veridata Username cannot be defined as an existing user in the existing MySQL database.
- The user responsible for creating a new MySQL Veridata repository user in an existing MySQL instance needs to have all privileges across all databases and tables. This

includes the ability to grant privileges to other MySQL users and allow connections from any IP address. The following is a sample SQL script for creating such a user:

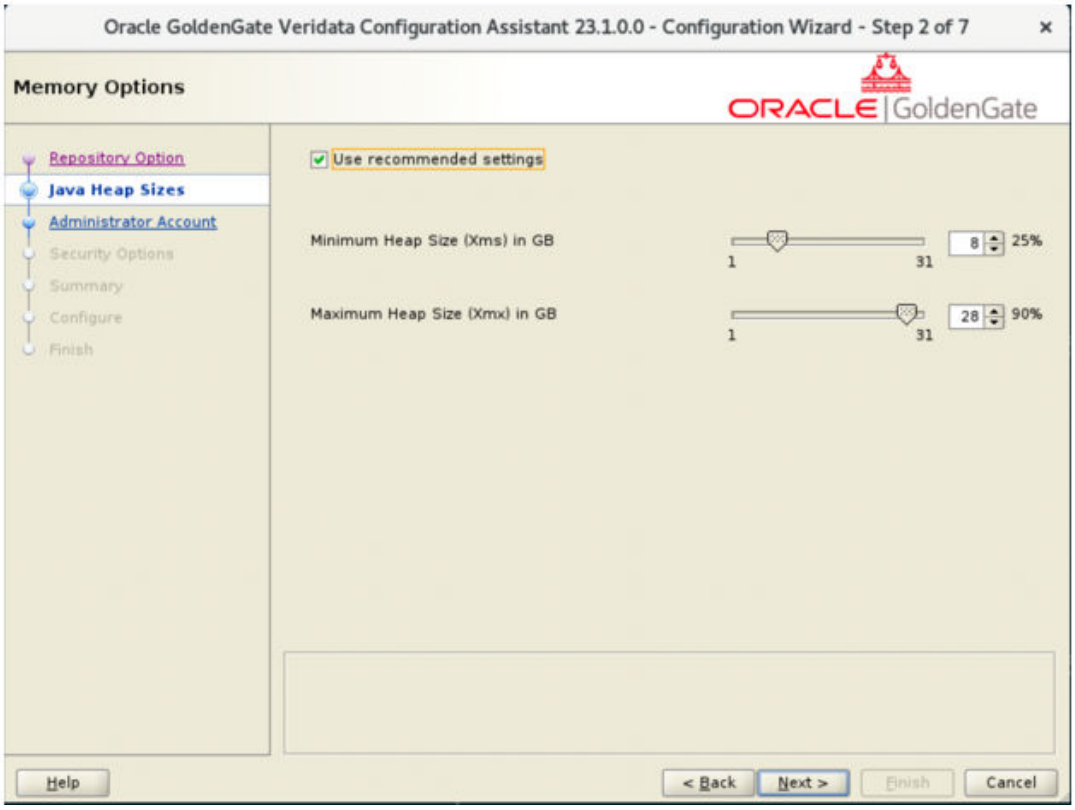
```
CREATE USER 'superuser'@'%' IDENTIFIED BY 'superpassword';
GRANT ALL PRIVILEGES ON *.* TO 'superuser'@'%' WITH GRANT OPTION;
FLUSH PRIVILEGES;
SHOW GRANTS FOR 'superuser'@'%';
```

Figure 3-8 Repository Creation Page for an Existing MySQL



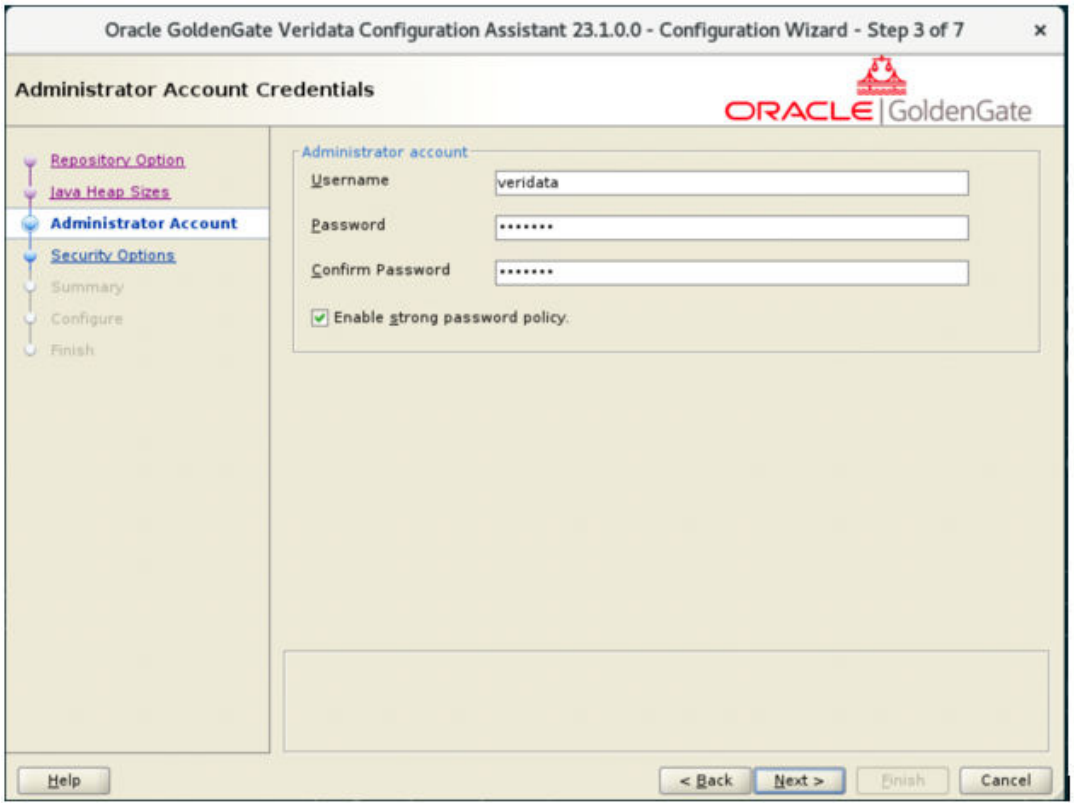
3. In the **Memory Options** page, for the **Java Heap Sizes**, select **Use recommended settings** (or modify according to your requirements) and click **Next**.

Figure 3-9 Memory Options



4. In the **Administrator Account Credentials** page, enter the Veridata Administrator **Username**, **Password**, and **Confirm Password** and click **Next**.

Figure 3-10 Administrator Account Credentials



5. The *Security Options* step of the Oracle GoldenGate Veridata Configuration Assistant allows users to configure the security settings for the Web User Interface (UI) by selecting encryption options, certificates, and communication ports. The primary function of this screen is to secure the connection between the client and the server using SSL/TLS protocols. This point outlines each of the available options and describes how to configure them.

   a. **SSL / TLS Security for Web UI**

      - **Purpose**: Enabling this option ensures that communication between the Web UI and users is encrypted using SSL (Secure Sockets Layer) or TLS (Transport Layer Security).
      - **Default Setting**: This checkbox is enabled by default, meaning SSL/TLS is turned on.
      - **Implications**: If disabled, the communication will not be encrypted, making it vulnerable to eavesdropping and other security threats. You can opt in to disable SSL/TLS for faster and simpler deployment in a test environment.

> **Note:**
>
> The SSL / TLS Security for Web requires 8830 port to be open apart from the selected port. If 8830 the port is not open, then Oracle GoldenGate Veridata server does not start after the installation. In such a case, do the following after installation:
>
> i. Update the following property in `<VERIDATA_HOME>/config/oggvdt_cainput.properties server.port=8830` to an available port.
>
> ii. Start the Oracle GoldenGate Veridata server from `<VERIDATA_HOME>/bin`: `./run.sh`

b. **Certificate Configuration**
To secure the Web UI, Oracle GoldenGate Veridata uses X.509 certificates. The configuration wizard provides two ways to specify certificates outlined as follows:

i. **Use Veridata Self-Signed Certificate**:

- **Purpose**: Use this option if you want Veridata to automatically generate a self-signed SSL certificate.
- **Pros**:
  - Quick and easy setup with no need for external certificate files
  - Useful in non-production environments or internal use cases where a trusted Certificate Authority (CA) is not necessary.
- **Cons**:
  - Self-signed certificates are not trusted by web browsers without manual intervention.
  - Not suitable for production environments where certificates must be validated by a third-party CA.

ii. **Upload Custom PEM Files**

- **Purpose**: This option allows users to specify their own SSL certificate, private key, and CA certificate files (in PEM format).
- **Required Files**: This is the public certificate (usually with a `.pem` or `.crt` extension) that will be used to identify the server to clients.
  - **File Input Field**: Certificate
  - **Example Path**: `/scratch/vdt23c/web/certificate.pem`
- **CA Certificate File**: This file contains the certificate(s) of the Certificate Authority (CA) that signed the server certificate. The CA file is used to verify that the server certificate is trusted.
  - **File Input Field**: CA Certificate File
  - **Example Path:** `/scratch/vdt23c/web/ca.pem`
- **When to Use**: This option is best suited for production environments where an official SSL certificate issued by a trusted CA (such as Let's Encrypt, DigiCert, or GoDaddy) is required.

c. **Port Configuration**
Oracle GoldenGate Veridata operates over specific ports, and users can define whether to use the default port or a custom one.

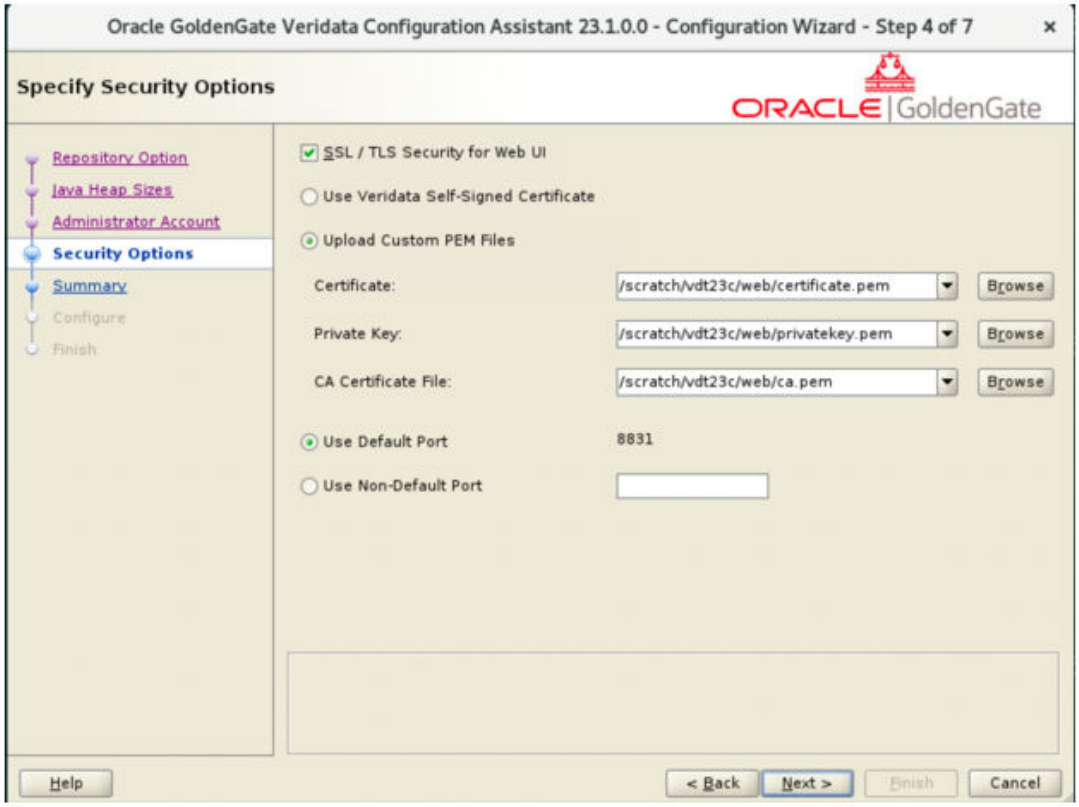
i. **Use Default Port (8831)**

- **Purpose**: The default port for SSL/TLS connections to the Veridata Web UI is 8831.
- **When to Use**
  - Suitable for environments where port 8831 is open and available.
  - Easier to configure, especially in less restrictive network environments.

ii. **Use Non-Default Port**

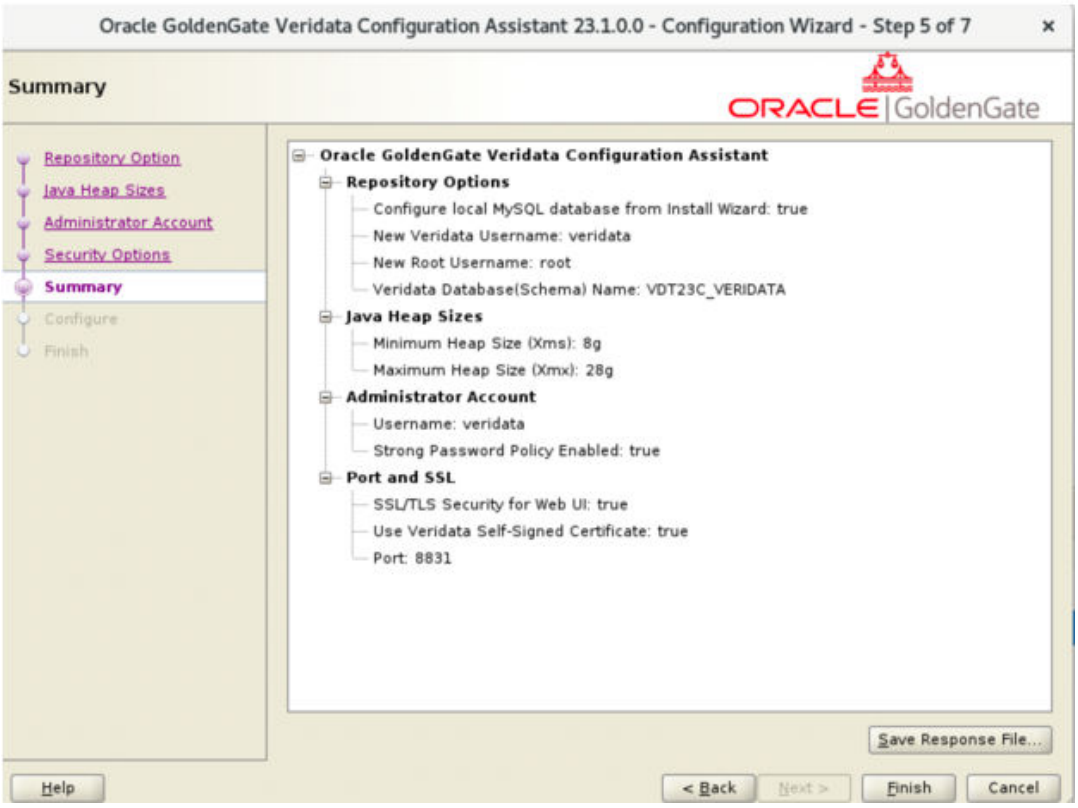
- **Purpose**: Allows users to specify a custom port for Veridata to use for SSL/TLS traffic instead of the default 8831.
- **When to Use**
  - In cases where the default port (8831) is blocked or already in use.
  - When a specific port policy is enforced by your IT or security team (for example, using port 443 for HTTPS traffic).
  - **Configuration**: Enter the custom port number into the input field provided.

Figure 3-11 Security Options



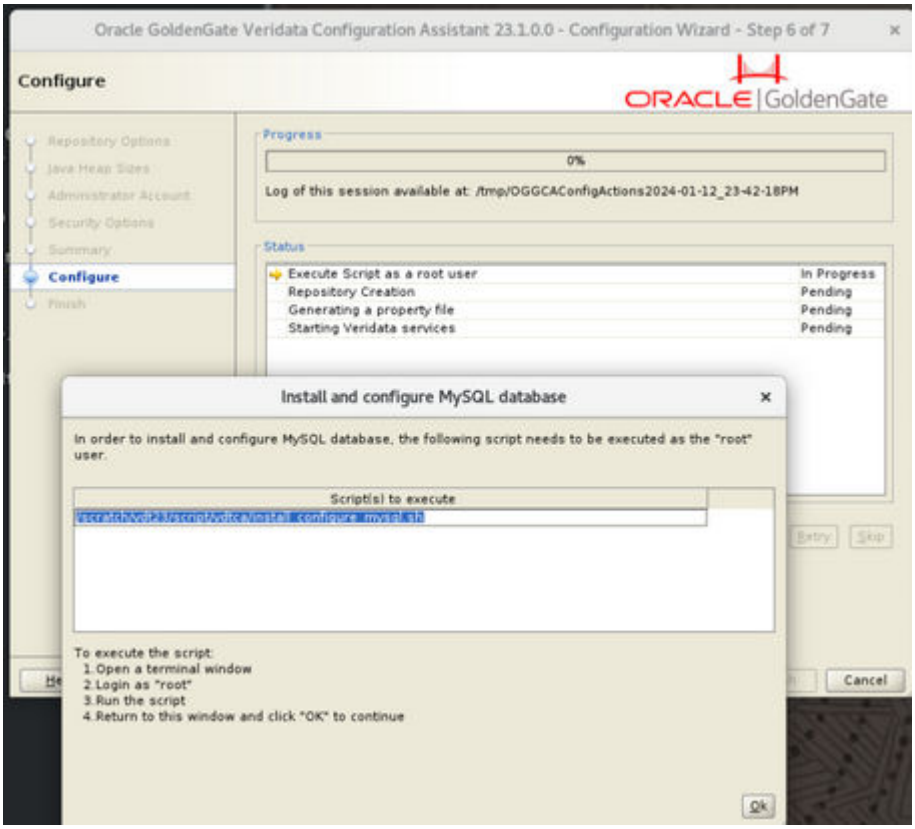
6. Review the configuration summary on the **Summary** page. Click **Save Response File** to save the respose file if needed.

Figure 3-12 Conifuguration Assistant Summary



7. Open a terminal and run `sudo su` to change to `root`. After you see the following screen, run the script as root.

Figure 3-13 Configure root



> **Note:**
>
> The root script should typically take 10-15 seconds.

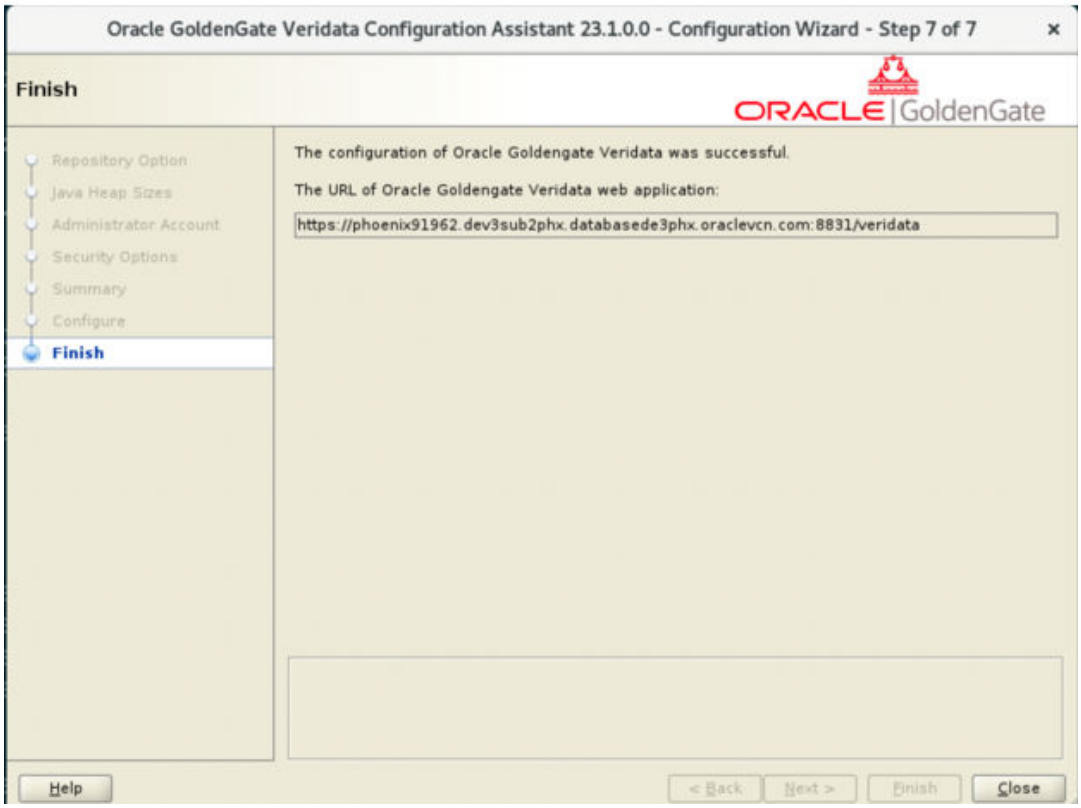
If the `install_configure_mysql.sh` script was executed successfully, the console will display the following information:

- The Configuration Assistant will set the root user password as what you entered in the next stage of veridata repository creation.
- MySQL has been installed and started.
- Please return to the Configuration Assistant and click **OK** to continue.

  Then follow the console instruction, return to the Configuration Assistant and click **OK** to continue.

8. After a successful configuration, the **Finish** page displays the URL of Oracle Goldengate Veridata web application. Input the URL to a web browser to verify that the Oracle GoldenGate Veridata server is running.

Figure 3-14 Configuration Complete



3.3 Installing and Configuring Source and Target Agents

From the agent directory, install and configure source and target agents.

- About Oracle GoldenGate Veridata Agent Deployment Scripts
- Configuring Oracle GoldenGate Veridata Agent
- Starting the Oracle GoldenGate Veridata Agent
- Using the Oracle GoldenGate Veridata Agent Deployment Script in Debug Mode
- Reloading the Logging Properties of the Veridata Agent

3.3.1 About Oracle GoldenGate Veridata Agent Deployment Scripts

After installing Oracle GoldenGate Veridata Agent using Oracle Universal Installer, you must deploy the agent to a non Oracle Home location and configure the agent before running comparison jobs using a deployment script provided in the installation.

The following table describes the directories and the variables that are used when referring to those directories in this section.

Table 3-1 Directories in an Oracle GoldenGate Veridata Installation

Directory Variable	Directory Path
VERIDATA_HOME	This is the home directory where Oracle GoldenGate Veridata is installed. For example, *VERIDATA_HOME* in this documentation is `/scratch/vdt23c`.
VERIDATA_AGENT_HOME	*VERIDATA_HOME*`/agent`. This is the location where Oracle GoldenGate Veridata agent is installed.
AGENT_DEPLOY_LOCATION	This is the location where the Oracle GoldenGate Veridata Agent is deployed. Note that this location should be outside the *VERIDATA_HOME*.

The `agent_config.sh` script located in the `VERIDATA_AGENT_HOME` directory is used for deploying the Oracle GoldenGate Veridata Agent.

Syntax:

```
export JAVA_HOME=/usr/java/jdk17
cd /scratch/vdt23c/agent
./agent_config.sh /scratch/agent1
./agent_config.sh /scratch/agent2
cd /scratch/agent1
cp sample_properties/agent.properties.oracle agent.properties
(EDIT agent.properties per documentation)
./agent.sh start agent.properties
```

The *AGENT_DEPLOY_LOCATION* can be an absolute path or a path relative to the location from where the script is running.

> **Note:**
>
> You must deploy the Oracle GoldenGate Veridata Agent to a directory outside *VERIDATA_HOME*.

3.3.2 Configuring Oracle GoldenGate Veridata Agent

You must configure the Oracle GoldenGate Veridata Agent to use your database.

1. Go to the agent deployed location *AGENT_DEPLOY_LOCATION*.
   There is an `agent.properties.sample` file in this directory that contains database related properties like JDBC URL and driver.
2. Copy the `agent.properties.sample` file and rename to `agent.properties`.
3. Most of the properties defined in the file have default values. However, you must update the following properties:
   a. The `server.port` property is the port where the Oracle GoldenGate Veridata Agent listens for connection requests.

      `server.port= <Port Number>`

      For example, *<Port Number>* = `7862`

b. The `database.url` specifies the JDBC connection URL for the database. Samples for all supported databases are provided in the file.

   `database.url=<Database URL>`

   For example, `<Database URL>=jdbc:oracle:thin:@localhost:1521:orcl`.

c. The `server.jdbcDriver` property specifies the list of JDBC driver JAR files. Sample lists for the supported databases are provided in the file.
   Use the driver corresponding to the `database.url` in the preceding step.

   `server.jdbcDriver=<JDBC Driver>`

   For example, `<JDBC Driver>`= `ojdbc11-23.2.0.0.jar`.

4. Oracle, MySQL, PostgreSQL, SQL Server, DB2 and Sybase drivers are available in *VERIDATA_HOME*`/agent/drivers`. For any other drivers you want to use, copy the driver to *VERIDATA_HOME*`/agent/drivers`. For more information, see Agent Parameters - Connections.

After the Oracle GoldenGate Veridata agent is installed, you need to configure 2-way SSL. See Configuring Workflow for Two-Way SSL in Oracle GoldenGate Veridata 23c.

3.3.3 Starting the Oracle GoldenGate Veridata Agent

Go to the agent deployment location *AGENT_DEPLOY_LOCATION* and run following command.

```
$ ./agent.sh start agent.properties
```

Where *agent.properties* is the properties file that contains your database properties.

3.3.4 Using the Oracle GoldenGate Veridata Agent Deployment Script in Debug Mode

For debugging issues with the Veridata Agent deployment, run the `agent_config` script with an additional command line argument as follows:

Syntax:

```
<VERIDATA_AGENT_HOME>/agent_config.sh AGENT_DEPLOY_LOCATION true
```

When this option is `true`, the debug logs are printed on the screen.

3.3.5 Reloading the Logging Properties of the Veridata Agent

You can reload logging information from the *VERIDATA_AGENT_HOME*`/config/odl.xml` configuration file to a running agent by using the `reloadLog` option. The changes in the `odl.xml` file are put into effect on the agent. The agent must be running for this command to work.

1. Open the command prompt and navigate to the directory where the agent is installed.
2. Go to *<AGENT_DEPLOY_LOCATION>* and run the following command:

   ```
   ./agent.sh reloadLog
   ```

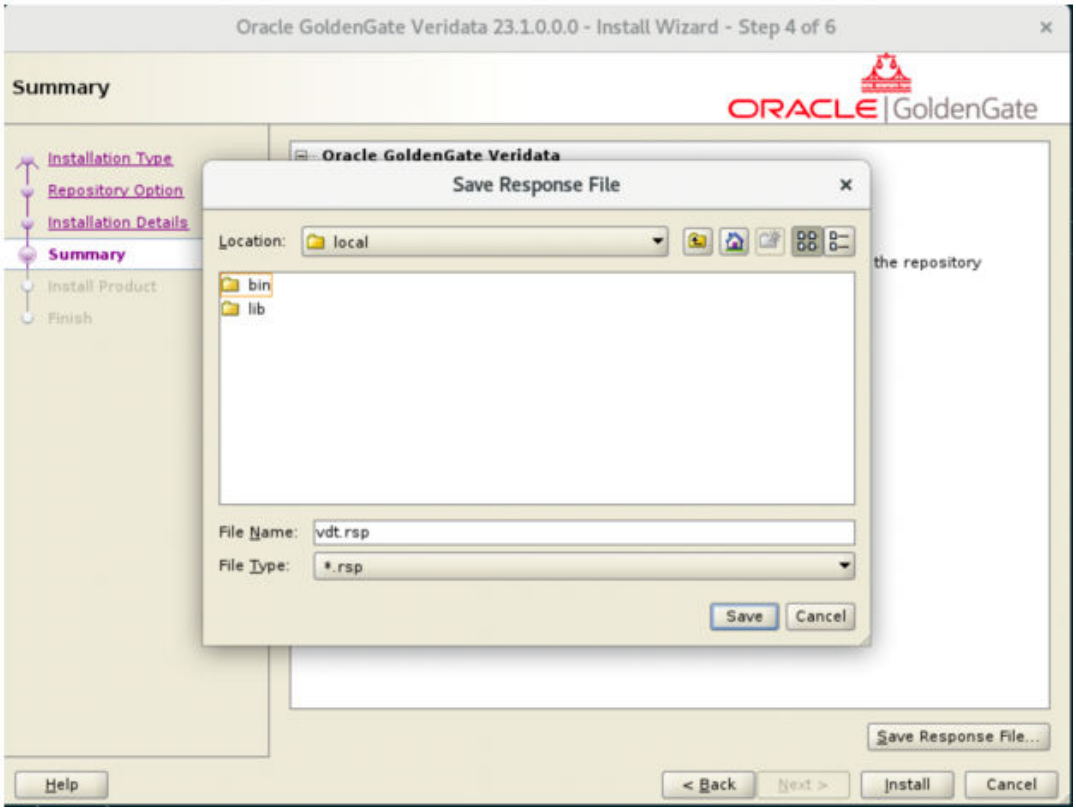
If agent properties are stored in a custom file other than agent.properties, then run the following command:

```
./agent.sh reloadLog <AGENT_PROPERTIES_FILENAME>
```

3.4 Silent Installation

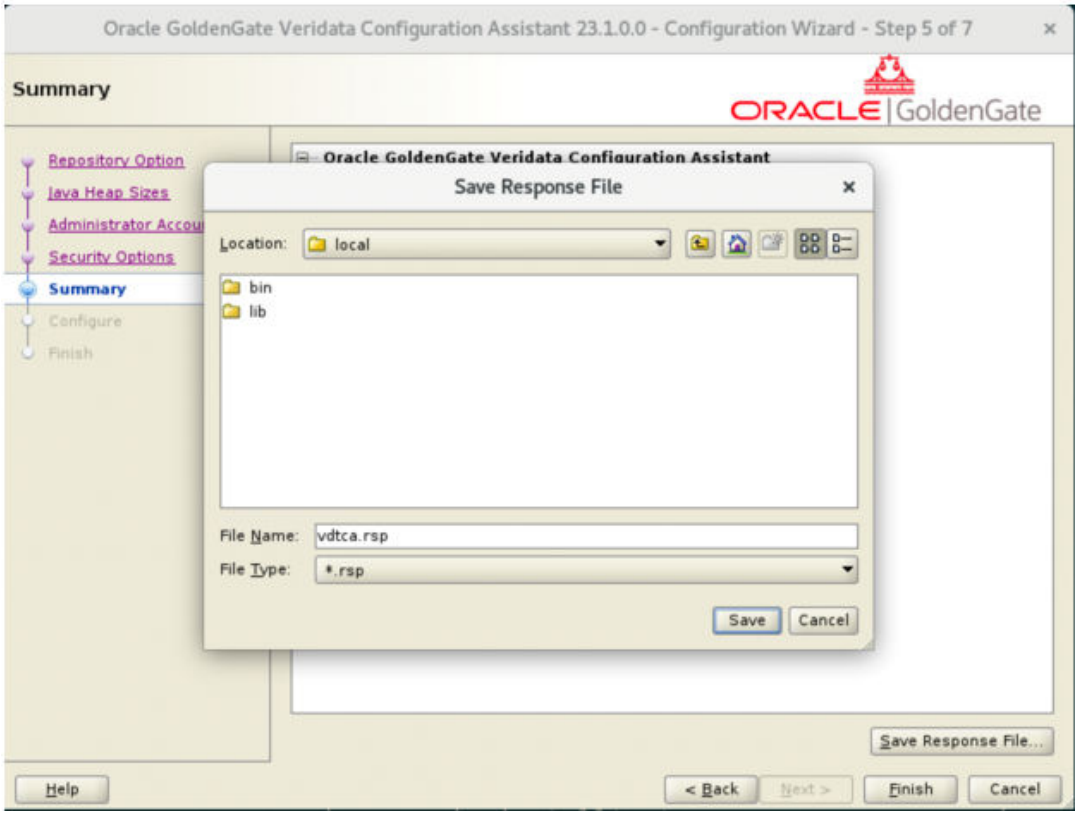
1. In the **Summary** page of the Installer, click **Save Response File** to save the response file. The default name of the Installer's response file is `vdt.rsp`.

Figure 3-15 Silent Installation - Save Response



2. In the **Summary** page of the Configuration Assistant, click **Save Response File** to save the response file. The default name of the Configuration Assistant's response file is `vdtca.rsp`.

Figure 3-16 Silent Install - Summary Page



3. The following is the command to make a silent installation of the Installer: `./fbo_oggvdt_linux_services_shiphome/Disk1/runInstaller -silent -responseFile /scratch/vdt.rsp`. If the silent installation was successful, the console will display the following information.

```
The installation of Oracle Veridata Services was successful.
Please check '/scratch/oracle/app/oracle/product/oraInventory/logs/
silentInstall2024-10-09_08-18-05PM.log' for more details.
Successfully Setup Software.
```

4. The following is the command to make a silent installation of the Configuration Assistant: `./<VERIDATA_HOME>/bin/vdtca.sh -silent -responseFile /scratch/vdtca.rsp` If the silent installation was successful, the console will display the following information: `Successfully Setup Software`.

3.5 Uninstalling Oracle GoldenGate Veridata

1. Run the following script to uninstall:
   `./<VERIDATA_HOME>/deinstall/deinstall.sh`

   The uninstall script first shuts down the Web Server, then shuts down the MySQL server. However, the uninstall script does not shut down the MySQL server and displays the following warning message if it cannot detect whether the MySQL server is up and running: `Unable to detect whether the MySQL server is up and running. Please manually check and shut it down if running.`

2. If the uninstall script does not stop the MySQL server, please manually stop it by using the MySQL command listed below. The MySQL command will prompt a password input. Use the root password you entered in the Configuration Assistant **Repository Options** step.

`<VERIDATA_HOME>/mysql-commercial-8.0.34-linux-glibc2.17-x86_64-minimal/bin/mysqladmin -u root -p shutdown`.
Another way to manually stop the MySQL instance process is using the Linux OS command: `kill -9 <mysql-instance-pid>`.

3. If the folders are located within `mysql-commercial-8.0.34-linux-glibc2.17-x86_64-minimal`, then take a backup of the folders data and the mysql-files from `mysql-commercial-8.0.34-linux-glibc2.17-x86_64-minimal`directory.
4. Remove the remaining MySQL-related directory using OS root user privileges: `sudo rm -rf <VERIDATA_HOME>/mysql-commercial-8.0.34-linux-glibc2.17-x86_64-minimal`

4
Get Started

4.1 Accessing Oracle GoldenGate Veridata Web User Interface

To protect data and comparison configurations, Oracle GoldenGate Veridata has security roles. Before attempting to use the Oracle GoldenGate Veridata Web User Interface, you should confirm which role has been granted to you by the Oracle GoldenGate Veridata Administrator. For more information, see Configuring User Groups.

To connect to the Veridata Web User Interface, open a web browser and type the following address. For example, `http://hostname/ipaddress:port/veridata` or `https://hostname/ipaddress:port/veridata` (`https` in case SSL / TLS Security for Web is selected while installation).

Where:

`hostname` is the name of the system where Oracle GoldenGate Veridata Server is installed and *`port`* is the port number where it is running.

> **Note:**
>
> Oracle GoldenGate Veridata 23c is designed to run optimally at HD resolution 1920x1080 and minimally 1342 x 672.

4.2 Home Page

The Oracle GoldenGate Veridata Home page comprises the following:

- **Left Navigation Pane**: Menu to all of the following Oracle GoldenGate Veridata objects:

  > **Note:**
  >
  > Users with less-privileged role may see only a subset of these objects. For more information, see Configuring User Groups.

  - Connections
  - Groups and Compare Pairs
  - Jobs
  - Monitor Jobs
  - Running Jobs

  - Profiles
  - Users
  - Utilities
- **Getting Started**: Comprises information about all the Oracle GoldenGate Veridata Objects and the tasks you can perform on the respective pages.
- **Dashboard**: The Dashboard area lists the following:
  - Compare Pair Status of Running Jobs
  - Compare Pair Status of Completed Jobs
  - Recently Completed Jobs
  - Favorites
- **Quick Guide**: Lists details of Oracle GoldenGate Veridata documentation.

The Home page also comprises the following:

- About
- Help
- User Menu

4.2.1 About

Click the **About** icon to get information about the Oracle GoldenGate Veridata version installed.

4.2.2 Help

Click the **Help** button to access the Oracle GoldenGate Veridata documentation.

4.2.3 User Menu

You can edit user account details, such as Description, Email ID, and Reset Password in the **My Accounts** page.

To access the **My Account** page, click **User Menu**, and then select **My Account**. The My Account page details the currently-logged-in user information, the Group that user belongs to.

In the **Preferences** page, you can set the time for refreshing any of the Monitoring pages, such as Completed Jobs, Running Jobs, or Repair Jobs. For example, if you (logged-in user) set the **Auto Refresh Interval** to 40, these pages will automatically refresh after 40 seconds. The default value is set to 30. To access the **Preferences** page, click **User Menu**, and then select **Preferences**.

To sign out off Oracle GoldenGate Veridata, click **User Menu**, and then select **Sign Out.**

5
Manage

5.1 Connections

To get started with Oracle GoldenGate Veridata, you must define a connection to the source and target databases that contain the data that you want to compare. Oracle GoldenGate Veridata Server uses the connection information to communicate with Oracle GoldenGate Veridata Agent.

A connection is defined by:

- A host where Oracle GoldenGate Veridata Agent is running.
- The port number for Oracle GoldenGate Veridata Agent on that host.
- The datasource that is associated with this agent.

Connections are managed from the **Connections** page in the navigation pane.

All connections that exist within the Oracle GoldenGate Veridata repository are shown in the list on this page.

> **Note:**
>
> Only users with Adminstrator and Super User roles can perform the following tasks:
>
> - Create Connections
> - Edit Connections
> - Delete Connections

5.1.1 Creating a Connection

To create a Connection:

1. Click **Connections** in the navigation pane.

On the **Connections** page that opens, enter the following parameters:

2. Enter a **Name** and **Description** for the new connection and click **Next**.
3. Enter the following parameters on the **Agent Connection** page:
   - **Host Name**: The DNS (Domain Name Server) host name or IP address of the host on which the Oracle GoldenGate Veridata Agent is installed.
   - **Port**: The port number that is assigned to the agent (or the Manager process, if a C-agent). To find out the port number of a Java agent, view the `server.port` parameter in the `agent.properties` file within the agent's installation directory. To find out the port number for a C-agent Manager, run the GGSCI program from the agent's installation directory, and then use the `INFO MANAGER` command.
   - **Database**: This is an optional field. You can click on **Verify** without selecting the datasource type. The correct datasource is identified and displayed after the verification is done :
     - Oracle
     - NSK
     - SQL Server
     - Sybase
     - DB2
     - Hive
     - Teradata
     - MySQL
     - PostgreSQL
   - **Use SSL for communication**: Select the **Use SSL for communication** checkbox for secure communication between the Veridata agent and the server.

     This option is is enabled only of you have set the `server.useSsl` parameter as true, while configuring SSL communication. See Parameters for Configuring SSL Communication.
   - A user name and password for connecting to the datasource (if required by the database).
   - (Optional) A separate user can be configured for executing repair operations at the target database. This user needs permission to update as well as to query the tables.
4. Click **Test Connection** to confirm that the connection is. If you have selected the **Use SSL for communication** checkbox in the previous screen, then SSL will be used for verifying the datasource connection.

5.1.2 Editing a Connection

1. Click **Connections** in the navigation pane.
2. In the connections list, click the edit (pencil) icon next to the connection that you want to edit.

   Alternatively,

   Click the connection name from the connections list, to display the **Connection Details** and **Connection Configuration** sections.

To edit the Connection Details:

1. Click **Edit** in **the Connection Details** section, to edit the description, Agent or Datasource connection details.
2. Click **Save** to save your edits.

To edit the Connection Configuration:
This tab modifies the behavior of the agent.

A check mark under **Use Default** indicates that a parameter is set to the default value.

The current setting is shown under **Value**.

If the parameter is a toggle, a check mark under **Value** indicates that it is enabled.

To change a setting

1. Clear the **Use Default** check box.
2. Edit Value.
3. Click **Save**.

Parameter Details

- **Agent Message Timeout**: Specifies a time interval, in seconds, after which the Server abends, if it has not received a message from the Agent.
- **Truncate Trailing Spaces When Comparing Values**: Space (U+0020) and Ideographic Space (U+3000) are the truncate targets. This parameter works only for String or Binary columns. You can configure the truncated length with the `truncate_spaces_len` entry in the `veridata.cfg` file of the Oracle GoldenGate Veridata Server. This parameter also truncates the column padding character if the padding character is one of the targe spaces. This parameter does not trim trailing spaces on LOB data.
- **Compare Fetch Size**: Sets the number of rows that are fetched at once for the initial comparison and for the confirm out of sync using batch or join. Increasing the batch size may increase throughput, as compared to standard database access.

  Exclusively, for the Oracle and PostgreSQL databases, the default value is 1000 and it fetches a batch size of 1000 rows; the minimum value that you can enter is 100. Any lesser value that you enter automatically changes to 100. The maximum value is 100000. For databases other than Oracle and PostgreSQL (SQL Server, Sybase, or DB2), the default value is 0, minimum is 0, and maximum value is 1000.

5.1.3 Deleting a Connection

1. To delete a connection, click the delete (trach can) icon.
2. Click **Delete** to confirm the delete action.

Alternatively,

1. Click on a connection in the connections list, click **Delete** in the **Connection Details** section.
2. Click **Proceed** to confirm the delete action.

> **Note:**
>
> Before deleting a connection, you must unlink it from any groups and jobs to which it is linked, or delete the group or job, if appropriate.

5.2 Groups and Compare Pairs

5.2.1 Group Details

Groups are logical containers for one or more Compare Pairs. They help you to organize and partition large or diverse sets of data into more manageable units. Groups are linked to jobs when jobs are created. Any group can be linked to one or more jobs, allowing you complete control over how and when data is compared.

> **Note:**
>
> A Group is associated with a set of connections to the source and target data. Before creating a group, you must create these connections.

The **Group Details** section of the **Groups and Compare Pairs** page lists the following details of the selected Group. You can also see the list of Compare Pairs associated to that Group.

- **Name**
- **Description**
- **Source Connection**
- **Target Connection**

You can modify the **Source Connection** and the **Target Connection** of the Group and click **Save** to save to Group details.

- Creating Groups and Compare Pairs
- Cloning a Group
- Deleting a Group

5.2.1.1 Creating Groups and Compare Pairs

To create Groups and Compare Pairs:

1. Click **Groups and Compare Pairs**, and then click **Create** to display the **Groups Details** page.
2. Enter the following details for the Group name and description **Name** and **Description**.
3. Select **Source Connection** and **Target Connection** for the Group. See Connections.
   - **Source Catalog**: The source metadata catalog or database.
   - **Source Schema**: The owner of the source database objects that are to be compared.
   - **Target Catalog**: The target metadata catalog or database.
   - **Target Schema**: The owner of the target database objects that are to be compared.
4. Click **Next** to display the **Mapping Rules** section. By default, **Mapping Rules** is toggled on. Mapping Rules help map source and target tables automatically based on rules set. If

you do not want to apply Mapping rules, then you can turn off this switch. Let's consider the Mapping Rules switch is toggled off.

5. In the **Mapping Rules** section, the following options are displayed **All**, **Mapped**, and **Unmapped**.
   - **All**: Lists all tables or files that are contained by the specified datasources. The Source tables are already listed. You can select the Target table from the Target drop-down list for Column Mapping.
   - **Mapped**: Shows only previously mapped tables or files.
   - **UnMapped**: Shows only tables or files not mapped in any other mapping.

- Mapping Rules
- Mapping
- Preview and Generate Compare Pairs

5.2.1.1.1 Mapping Rules

To set the Mapping Rules for the Groups and Compare Pairs:

1. In the **Mapping Rules** section, select one of the following methods for matching source table names to target table names. By default, **Mapping Rules** is toggled.
   - **Using Exact Names**
     This pattern matches names character-for-character, so ensure that each source and target name are identical. As an example, this mapping rule is useful for comparing production and failover databases. By, default the **Using Exact Names** option is selected. Select **Using Exact Names** if you want to map the tables defined at database. For example, if there are 10 partitions in Source Table matching 10 table partitions in Target Table with same table names, then 10 compare pairs are created. If no table exists then, one compare-pair per table gets generated. After all partitions are mapped, details of compare-pairs generated are displayed in Preview Tab.
   - **Using Wildcard Pattern. (Enclose within single % on either side)**
     To use this method, supply a wildcard string in **Source Pattern** and **Target Pattern Like**, and select either Like or **Not Like** options (for both Source Pattern as well as Target Pattern) that include the percent symbol (%) as the wildcard. A % in the target matches the text that is matched by the % in the source. By default, the **Like** option is selected for both the source and target tables. You can select **Not Like** if needed during the compare-pair generation.
     – Example:

       Assume source tables of:

       ```
       SOURCE_TABLE_1
       SOURCE_TABLE_2
       MY_SOURCE_TABLE_1
       MY_SOURCE_TABLE_2
       DUMMY_TABLE
       ```

       Assume target tables of:

       ```
       TARGET_TABLE_1
       TARGET_TABLE_2
       MY_TARGET_TABLE_1
       MY_TARGET_TABLE_2
       DUMMY_TABLE
       MY_DUMMY_TABLE
       ```

Some possible **Like** pattern matches are:

Table 5-1 Examples for Mapping Rules Using Wild Card Option % - Like option selected

Source Pattern	Target Pattern	Matches
%SOURCE_%	%TARGET_%	SOURCE_TABLE1=TARGET_TABLE1, SOURCE_TABLE2=TARGET_TABLE2
%MY_SOURCE_%	%MY_TARGET_%	MY_SOURCE_TABLE1=MY_TARGET_TABLE1, MY_SOURCE_TABLE2=MYTARGET_TABLE2
%SOURCE_%	%MY_TARGET_%	Table1=Table2 TABLE2=TABLE2
%DUMMY_TABLE%	%_TABLE%	DUMMY_TABLE=DUMMY_TABLE
%DUMMY_TABLE%	TARGET_%_	None

Table 5-2 Examples for Mapping Rules Using Wild Card Option % - Not Like option selected

Source Pattern	Target Pattern	Matches
%SOURCE_%	%TARGET_%	DUMMY_TABLE=DUMMY_TABLE

  - The wildcard resolution is case-sensitive.

- **Include Views**: The **Include Views** check box is selected by default. In the Mapping page, you can select the compare pairs based on the Mapping rules. This also has the View Table pair if you include views in the mapping rules page. Mapping rules are ways to ease mapping or to automatically create mapping between source and target database based on selected rules. If you do not want to include Views, then you can deselect the check box.
- **Include Table Partitions**: The **Include Table Partitions** check box is selected by default. This option allows you to include all the table partitions (when applicable to the database) while generating compare pairs. If you do not want to include table partitions, then you can deselect the check box.

2. After you have selected the Mapping Rules, click **Next** to display the **Mapping** section.

5.2.1.1.2 Mapping

By default, the **Mapping** section shows all of the source and target objects that are contained by the specified datasources. You can filter the list at any time in the process of creating compare pairs. For example, after you finish mapping several compare pairs, it might be helpful to reduce the size of the list by using the filter to show only unmapped objects. This saves you navigation time.

To filter the list of mappings, click one of the following options:

- **All**: Lists all tables or files that are contained by the specified datasources. To view all the Source and Target tales, select the **All** option. However, ensure to give this Compare Pair a different name.
- **Mapped**: Lists only previously mapped tables or files.
- **Unmapped**: Shows only tables or files not mapped in any other compare pair.

You can also enter a search string in the Search box to filter the mappings list.

To map tables:

1. Select the **Source** table. In case of the All option selected, all the Source and Target tables are selected by default. You may want to deselect the tables that you do not want to be mapped.
2. From the **Target** drop-down list, select the target table.
3. (Optional) You can also toggle Automatic Row Partition to enable partitioning of multiple rows.
4. After the mapping is complete, click **Next** to go to the **Preview** section and review the configurations of the **New Compare Pairs**. The following details are displayed:
    - **Compare Pair Name**
    - **Source Table**
    - **Target Table**
    - **Profile**

- Automatic Row Partition
  If you want to split the large table comparison into multiple partitions, then you can use Automatic Row Partition. The partitioning is possible only when both the source database and target database are Oracle.

5.2.1.1.2.1 Automatic Row Partition

If you want to split the large table comparison into multiple partitions, then you can use Automatic Row Partition. The partitioning is possible only when both the source database and target database are Oracle.

The automatic row partition creates configurable partitions and generates compare pairs for each generated partition.

To configure Automatic Row Partition:

1. On the **Mapping** section of the **Create Group and Compare Pair** (or **Edit Group and Compare Pair**) page, select the Source and Target tables.
2. Toggle **Automatic Row Partition** against the selected table pair. By default, the toggle option is disabled.
3. Select a numeric value between **2 to 100** in the text field before generating compare pair. The minimum value you can select is 2 and the maximum is 100. This value decides the number of compare pairs to be created. Each auto generated compare pair compares the subset of data from the selected source and target table.
4. After the required datasources are mapped, details of compare pairs generated are displayed in **Preview** section.

> **Note:**
>
> The **Automatic Row Partitions** check box is available only when you manually select the source and target tables in the Mapping section. You cannot edit compare pair names that are created using Automatic Row Partition.

5.2.1.1.3 Preview and Generate Compare Pairs

You can view compare pairs that can be generated by using the mapping in the **Preview** section. You can select the compare pairs in the Preview page. Based on the selection, you can generate the compare pairs.

1. If you want to modify any of the mappings, you can click **Back** to go back to the **Mapping Rules** section, modify the details. For example, from the currently selected **Using Exact Names**, if you want to switch to **Using Wildcard Pattern** and enter `emp%` in the **Source Pattern** and the **Target Pattern**, and select **Like**, and then click **Next** to display the **Retain Compare Pairs** dialog box. Click **Yes** if you want to retain the previously-selected compare pairs. If you click **No** and proceed further, the previously-selected compare pairs are not retained and Compare Pairs of the Employee tables are displayed in the **Preview** section.
2. After reviewing the Compare Pair details, click **Generate Compare Pairs**.

The compare pairs are generated and are listed in a tabular format. See Compare Pairs.

5.2.1.2 Cloning a Group

You can clone a Group from the **Groups and Compare Pairs** page.

1. To clone a Group, select a **Group** from the list, and click the 3 dots (…) and select **Clone** to display the **Enter Group details** dialog box.
2. Enter the **Group Name** and **Description** and click **Clone**.

A new Group is created with the same Compare Pair details of the Group you just cloned.

5.2.1.3 Deleting a Group

You can delete a Group from the **Groups and Compare Pairs** page.

1. To delete a Group, select a **Group** from the list, and click the 3 dots (…) and select **Delete**.
2. Click **Delete** in the confirmation dialog box to confirm deletion of the selecte **Group**.

A **Group deleted successfully** message is displayed after the Group is deleted.

5.2.2 Compare Pairs

A compare pair is the logical relationship between a source table or file and a target table or file for the purpose of comparing their data. Compare pairs are linked to groups. As a reason, all of the source and target objects that you configure into compare pairs for any given group must be accessible from the datasource connections that are associated with that group.

The **Groups and Compare Pairs** page lists the following details of the Compare Pairs created for that particular Group.

- **Enabled**: By default, all the compare pairs are enabled. You can toggle this switch to disable the selected Compare Pair. You can select the compare pairs to be included in the

Group by toggling the **Enabled** switch, and then click **Save** to save the Group and Compare Pairs details.

- **Compare Pair Name**: The default name format is `<source>=<target>`. If you hover the mouse cursor over a Compare Pair name, the actual source and target object names are displayed.
- **Source Table**: The source table in the Compare Pair.
- **Target Table**: The target table in the Compare Pair.
- **Delta Processing**: See Delta Processing
- **Validation Status**: By default, the Validation Status is **Never Validated**. See Validate Column Mapping. After you select the Compare Pair and click **Validate Column Mapping**, the **Validatation Status** of the selected Compare Pair gets updated to **Validated**. You can also select multiple Compare Pairs and click **Validate Column Mapping** to validate their status.
- **Actions**: You can delete the selected compare pairs by clicking the **Delete** icon. See Deleting a Compare Pair

Use the **Compare Pair** section on the **Groups and Compare Pairs** page to do the following:

- Add Compare Pairs
- Validate Column Mapping
- Enable Delta Processing
- Delete Compare Pairs
- Compare Pair Details

5.2.2.1 Compare Pair Details

The **Compare Pair Pair Details** section of the **Groups and Compare Pairs** page lists the following details of the selected Compare Pair. Click a Compare Pair to view the following Compare Pair details:

- **Source Table**: The source table or database.
- **Source Schema**: The owner of the source database objects that are to be compared.
- **Compare Pair Name**: This is either the default name format of `<source>=<target>` or a user-defined name. In either case, if you hover the mouse cursor over a Compare Pair name, the actual source and target object names are displayed.
- **Profile**: If a run profile exists for a compare pair, it is shown here. Otherwise, this field is blank and the default profile will be used during comparisons.
- **Target Table**: The target table or database.
- **Target Schema**: The owner of the target database objects that are to be compared.
- **Validation Status**: Shows whether or not the columns of the source and target objects are suitable for being compared, based on the results of any previous validation that was performed.

You can modify the **Source Table**, **Source Schema**, **Target Table**, **Target Schema** and update the **Profile** of the Compare Pair, and then click **Save** to save to Compare Pair details.

Click **Full Screen** to view the view details of the Compare Pair on full screen. To go back to the default view, click **Partial Screen**.

Click **Back to Compare Pairs List** to go back to the **Compare Pair** section of the **Groups and Compare Pairs** page.

For a selected Compare Pair, you can also perform the following from the Compare Pair Details section:

- Column Mappings
- Delta Processing
- Row Partitions
- Deleting a Compare Pair

5.2.2.1.1 Column Mappings

Use the **Column Mapping** tab to create or change a column mapping.

You must have the Administrator or Super User role to create or edit column mappings.

The **Column Mapping** tab lists the following Compare Pair details:

- **All**: Lists all tables or files that are contained by the specified datasources. Ensure to give this compare pair a different name.
- **Mapped**: Lists only previously-mapped columns.
- **Unmapped**: Lists only columns not mapped with any other column of the compare pair.
- **Source**: Details of the source table, such as Include, Name, and Data Type.
- **Target**: Details of the target table, such as Name, Data Type, Format, and Key.

- Validate Column Mapping
- Key Mapping Method

5.2.2.1.1.1 Validate Column Mapping

Validation is a preliminary test to determine whether or not the source and target table structures are compatible, and that they both have primary or unique key columns that match. You can perform a manual validation at any time. Oracle GoldenGate Veridata always performs a validation at runtime.

To validate column mapping, on the **Groups and Compare Pairs** page **> Compare Pairs** section, select a Compare Pair, click **More Actions**, and then select **Validate Column Mapping**. By default, the Validation Status is **Never Validated**.

The selected Compare Pair is validated and the following message is displayed: **Compare Pairs Validated Successfully**.

5.2.2.1.1.2 Key Mapping Method

There are two methods for mapping key columns and comparison columns:

- System Generated: Column mappings are configured automatically by Oracle GoldenGate Veridata at runtime based on current object metadata.
- User Defined: Column mappings are configured manually by an Oracle GoldenGate Veridata user who has the Administrator or Super User role.

All new compare pairs default to System Generated for key columns and for comparison columns. You can change to a different mapping method at any time.

How these methods apply to keys

- **System Generated**: If you know that the objects in a compare pair both contain a primary key or a unique index, you can leave the key mapping method set to the default of System Generated. The key columns will be mapped automatically. To map keys, Oracle GoldenGate Veridata finds all indexes on the source and target objects and tries to find a primary key on each one. If primary keys are not found, Oracle GoldenGate Veridata tries to use the smallest index (least number of columns), and then it maps the columns that have identical names and comparison formats. Any columns that cannot be matched are excluded from the configuration.
- **User Defined**: If an object has neither a primary key or unique key, you can use the User Defined method to map key columns manually, one by one. You can also use the user-defined method to override existing keys or indexes, but the columns that you select to use as a key must ensure the uniqueness of rows. Also avoid using source and target indexes that have different precision levels or other characteristics that can reduce the accuracy of row selection, especially in a heterogeneous environment.

How these methods apply to comparison columns

- **System Generated**: If the source and target comparison columns have the same names and comparison formats, you can leave the comparison column mapping method set to the default of System Generated. Oracle GoldenGate Veridata will map those columns automatically at runtime. Non-matching columns are excluded from the configuration. By default, this method includes all of the columns in a comparison. This method defaults to the hash comparison method. You can change the comparison method later by editing the compare pair.
- **User Defined**: Use this method to map source and target columns manually and to control the comparison method.

You can combine these methods to speed up the mapping process. If most of the column names support system-generated mapping, you can use it and then switch to the user-defined method to map the remaining columns, or to exclude columns from the comparison. For example, you can exclude columns if you know that their values never change or if you expect their values to be out-of-sync.

5.2.2.1.2 Delta Processing

This topic provides answers to typical questions about the delta processing feature that is available for all Oracle GoldenGate Veridata supported databases.

In Oracle GoldenGate Veridata, the source and target tables are configured using compare pairs, which are grouped and added to a job to run the comparison.

During the subsequent runs of a comparison job, the comparison of the tables can be performed based on what has changed in the tables from the previous job run; these jobs are Delta Processing Jobs.

- For all supported databases, you can use the delta processing performance feature if you are using server-side sorting.
- For the NonStop platform, Oracle GoldenGate Veridata finds a changed block by detecting a change in its Volume Sequence Number (VSN) since the time of the last comparison.
- For all platforms, the delta comparison column must contain a value that is modified every time the row is modified and this value must always keep increasing. For example, `TIMESTAMP` or `NUMBER`. For the Oracle database, `ROW_SCN` is a default delta column, which keeps on increasing at every delta run.

- The comparison of the tables can be performed based on what has changed during the subsequent runs of a comparison job.

With delta processing enabled, Oracle GoldenGate Veridata compares the difference between the 2 consecutive jobs.

To enable Delta Processing:

1. In the **Groups and Compare Pairs** page, click the **Delta Processing** tab under the **Compare Pairs Details** section.
2. Ensure that the column names are populated in the **Use Source Delta Column** and **Use Target Delta Column**. Note that you cannot edit the source and target queries.
3. From the **Edit Group Compare Pair** page also, you can enable delta processing by selecting the particular Compare Pair and by selecting the dropdown option to enable delta processing.

- What is Delta Processing?

5.2.2.1.2.1 What is Delta Processing?

How does it work on NonStop Platforms?

Oracle GoldenGate Veridata finds a changed block by detecting a change in its Volume Sequence Number (VSN) since the time of the last comparison. The VSN is a disk-specific change number that increments sequentially with each database operation that is performed on the data. Each time that a row changes, there is a change in the VSN of the disk block where the row resides.

There is no relationship between a VSN in a file on one disk and a VSN on another. Oracle GoldenGate Veridata tracks VSNs on a per-partition basis on the source and target disks and maintains its own correlations to perform accurate delta comparisons. Once you enable delta processing, it is used for all subsequent runs until you disable it again.

> **Note:**
>
> The first run of a compare pair always compares all of the rows in the source and target objects to establish an initial VSN state from which to evaluate deltas in future runs.

How does it work on all other Platforms?

Oracle GoldenGate Veridata compares a source database table to the target database table. The source and target tables are configured using compare pairs, which are grouped and added to a job to run the comparison (see Compare Pairs). When all the rows in the table are compared, it is a Full Comparison Job.

During the subsequent runs of a comparison job, the comparison of the tables can be performed based on what has changed in the tables from previous job run; these jobs are Delta Processing Jobs. Delta processing is usually performed on tables that contain a large number of rows so it is probable that in these tables there will be columns eligible for delta processing. The delta comparison column must contain a value that is modified every time that the row is modified and this value must always be increasing. Any data type that meets this requirement is supported. By default, the columns of the table that are mapped to Numeric or Timestamp comparison formats are supported. For example, `TIMESTAMP`, `TIMESTAMP_TZ`, and `NUMBER`.

The delta base is the value of the Delta Column on the basis of which the delta comparison was performed. Every time a comparison is run, a delta base value is captured. Depending on the number of delta comparison jobs performed, there can be multiple delta base values so a list of delta base values for the compare pair is generated. For example, the first time a Full Comparison is run and the maximum value of the Delta Column is the delta base, `DeltaBase-1`. A second Delta Processing Job run based on `DeltaBase-1` results in `DeltaBase-2` being captured again as the maximum of Delta Column. In the third run, you can use either `DeltaBase-1` or `DeltaBase-2` for the comparison or run a Full Comparison Job.

When Should I use Delta Processing?

Delta processing is suitable for use with very large Enscribe files and SQL tables that, otherwise, would take a long time to process. It does consume additional overhead, so it is probably not practical for use with smaller sets of data. Try running a test comparison without delta processing first. If, in your opinion, the compare pair takes too long to process, try running it again with delta processing enabled. If the delta-enabled run is significantly shorter than the first test, continue to use it. If there is only marginal improvement, it might be better to disable delta processing to prevent the added overhead. The performance gains of delta processing are in the initial comparison step of the run. Delta processing can cause the confirmation step to be longer if the source and target rows end up on different data blocks.

What Process Performs the Delta Processing?

The delta processing is performed by the Oracle GoldenGate Veridata Agent.

For NonStop platforms, the VSN information is retrieved by a privileged process named `vsnserv`. During the installation of Oracle GoldenGate Veridata Agent on a NonStop platform, `PROGID` was used for the `vsnserv` program to run as `SUPER.SUPER` to be able to read the file labels for this purpose.

For all other platforms, the delta processing queries the compare pair to retrieve the delta base values for both the source and target tables, which creates the column mapping.

What sorting method can be used with delta processing?

To use delta processing, you must enable server-side sorting by setting the sorting method to Server within the profile that is associated with the compare pair or the one that is associated with the job when you run it.

> **Note:**
>
> If you always will be using delta comparisons, then consider setting the sorting method to Server within the default Oracle GoldenGate Veridata profile. That way, nobody will forget to select the correct profile when the jobs are run.

What other important things should I know when using delta processing?

The delta processing mechanism can fail to detect an out-of-sync delete, if that delete was the only source row that was modified in a block, and if that delete did not get propagated to the target. In such a case, the block on the target that contains the relevant row does not get modified, so it is skipped by the target Veridata Agent during delta processing.

The delta value field is pre-populated in the column mapping UI, based on the query for retrieving the delta value.

Support Considerations

The following table shows the supported delta column types. The possible list of delta columns for the delta configuration UI is identified by reviewing the compare formats for the corresponding column-pairs in the source and target tables.

Veridata Comparison Format	Can it be Delta Column?
STRING	No
STRING_EN	No
TIMESTAMP	Yes
TIMESTAMP_TZ	Yes*
DATETIME	Yes
SMALLDATETIME	No
DATE	No
TIME	No
NUMBER	Yes
FLOAT	No
BINARY	Yes*
BLOB	No
CLOB	No
INTERVAL	No
DEC_FLOAT	No
BINARY_TIMESTAMP	Yes*
SYBTIME	No
CLOB_NFC	No
STRING_NFC	No

> **Note:**
>
> * denotes that Hive doesn't support the delta column. Supported data types for Hive are NUMBER, TIMESTAMP and DATETIME.

For Oracle Database, the ORA_ROWSCN pseudo delta column is supported, and selected by default when ROWDEPENDENCIS are enabled for that table.

For DB2 for i, z/OS, and LUW, columns having the GENERATED FOR EACH ROW ON UPDATE AS ROW CHANGE TIMESTAMP clause is selected by default.

For all supported databases, TIMESTAMP columns are shown in order first then followed by NUMERIC columns.

Only one column is supported for delta processing and is similar to:

```
SELECT column-names from table name where delta_column delta_condition
supplied_delta_value
```

For example:

```
select * from TableA where startdate >= '12-01-2012 21:24:00'
```

How do I enable or disable delta processing?

You can enable or disable delta processing for database compare pair from the Delta Processing tab and the Existing Compare Pairs tab of the Compare Pair Configuration page. In addition, you can use the Delta Processing Enabled check box on the Run Configuration page to toggle this feature.

How do I configure delta processing?

On the Column Mapping Configuration page, click the Delta Processing tab.By default, the options are automatically populated and the Enable Delta Processing check box is selected. You can disable the feature by clearing the check box, which renders all other options inactive.

You can use the defaults or change any of the following:

- the source or target columns for processing,
- whether to use a source or target query
- the source or target database query

After you have configured the delta processing for the compare pair, click **Save** to apply your changes.

How do I know that delta processing is being used?

When a compare pair is configured for delta processing, `Delta Enabled` text is displayed in the compare pair table of the Edit Group and Compare Pair Configuration pages.

Can I override delta processing when I run a job?

What if I perform maintenance on the tables or files for which I am using delta processing?

When you perform maintenance on objects in a compare pair that has delta processing enabled, the best practice is to disable delta processing for the next run so that Oracle GoldenGate Veridata compares all of the rows. You can disable delta processing at the compare pair level or as a job override. Starting again with a full comparison allows a new delta base state to be established and will make subsequent delta comparisons faster. Otherwise, delta processing could actually take longer than a complete comparison of all rows.

For example, if a 'FUP RELOAD' is performed on the source, but not on the target, it could cause delta processing to return a much larger number of rows from the source than from the target, based on the last delta state. The source rows that are returned would be rows that actually did not change. This happens because FUP moves records around and combines blocks, but does not change the data. However, the VSNs for the affected blocks will change. Oracle GoldenGate Veridata cannot detect that the reload was done since the last delta state. Thus, the next time that the VSN for a block changes, all of the rows in that block will be returned.

Conversely, on the target, no VSNs are changed for the corresponding data (because a reload was not done there), so those rows are not returned for delta processing. This anomaly will be resolved by the confirmation step, but this slows the overall comparison process because that step is much slower than the initial comparison step.

5.2.2.1.3 Row Partitions

You can specify which rows to include or not to include in a comparison by specifying a SQL predicate statement or an Enscribe partition.

Oracle GoldenGate Veridata supports selecting a subset of rows for comparison by row partioning (SQL predicate statement) or by Automatic Row Partition.

Using partitions allows you to compare source and target tables or files that have the same structure but a different number of rows. For example, you could compare a production table to a data warehouse table that may contain more rows because of historical data. The usage of partitions also speeds throughput by splitting the load into multiple processing streams.

1. Click the plus (+) icon in the **Source Partition** area. The **Source Partition** button is toggled on by default.
2. In the Name box, type a name for this partition, such as `emp`. Use one word that can include underscores, hyphens, and other standard keyboard special characters.
3. Enter a **SQL Predicate**. For example, `where emp_salary>1000`. The SQL Predicate Statement is a condition expression, where the `WHERE` clause is used to evaluate to a boolean value, either `true` or `false`. You can also edit or delete the SQL Predicate by clicking the Pencil or the Delete icons adjacent to it.
4. Similarly, enable **Target Partition**, and click **Copy from Source** to copy the SQL Predicate statement details from the source.

5.2.2.1.4 Deleting a Compare Pair

Use the Compare Pair Configuration page to delete compare pairs.

You must have the Administrator or Super User role to delete compare pairs.

1. To delete a Compare Pair, on the **Groups and Compare Pairs** page **> Compare Pairs** section, select a Compare Pair, click **More Actions**, and then select **Delete**.
2. Click **Delete** in the confirmation dialog box to confirmation deletion.

5.3 Profiles

A profile is a set of global processing parameters, each containing unique settings for a specific purpose. Oracle GoldenGate Veridata provides a default profile, but you can create your own profiles. You can create as many profiles as needed and associate them with any job or compare pair (to override the job profile; see **Compare Pairs**). You can override profile assignments at run time.

You must have the Administrator or Super User role to create, edit, or delete a profile.

You can create, edit, clone, delete, and view profile information from the **Profiles** page.

To create a Profile:

1. Click **Profiles** in the navigation pane.
2. On the **Create a Profile** page, click **Create**.
3. Enter the name and descriptoin for the new profile.
4. From the **Create new profile using copy of existing profile** drop-down, select an existing profile you want copy to create a new profile.

5. Click **Submit**

To edit a Profile:

1. Click the profile name or the corresponding edit (Pencil) icon.
2. On the **Profile Information** page, edit the required settings and click **Save**.

   See Editing Profile Settings.

To clone a Profile:

1. Click the copy icon next to the profile that you want to clone.
2. Enter a name and a description for the new profile. Click **Clone**. You can now see the cloned profile in the profiles list.

To delete a Profile:
Click the corresponding delete icon in the **Actions** column. Alternatively, click the profile name, then click **Delete** on the **Profile Information** page.

Change the default profile

Oracle GoldenGate provides a default profile (see Using the Default Profile) that is used for all of the compare pairs and jobs that are not linked to a user-defined profile. This profile is used automatically unless another profile is selected when creating a compare pair or a job, or when running a job. You can change the settings of the default profile by editing it.

- Editing Profile Settings
- Using the Default Profile

5.3.1 Editing Profile Settings

This section describes the parameters that you can edit for a profile.

1. **General**: Controls output options. You can set the following output options:
   - **Report In-sync rows to Report file**: Prints all the in-sync rows to the veridata report.
   - **Report In-sync after In-flight rows to Report file**: Print the in-flight rows that becomes in-sync after the Confirm Out-of-Sync phase.
   - **Out-of-Sync Output Format**: From the dropdown, select a format for the out-of-sync file that is generated. The default value is **Binary**. If you select **None**, there is no output file generated.
   - **Maximum size of each Out-of-Sync XML chunk (rows)**: You can set the maximum chunk size for the number of out-of-sync rows for the server-to-agent calls. Larger chunk size boosts perfomance by minimizing the server-to-agent calls, but utilizes more network bandwidth.
2. **Sorting Method**: Controls sorting method and memory management. Data is sorted to match keys (or a key specification) so that the correct source and target rows are compared.
   - **Sort Data Using**: Selecting the **Server** option gets and sorts the data on the server. Selecting the **Database** option uses the `order by` clause in the table, to sort the data during comparison.
   - **Temporary Storage Directory for Source Data**: Creates the sort file for the source data in the specified directory. Default location is the `/tmp` directory.
   - **Temporary Storage Directory for Target Data** Creates the sort file for the Target data in the specified directory. Default location is the `/tmp` directory.

3. **Initial Compare**: Controls parameters for the process that performs the initial compare step.
   - **Output Out-of-Sync Record Details to Report File**: Select this option to print all the out-of-sync records to an output report file.
   - **Terminate when Maximum records Out-of-Sync**: Set the number out-of-sync records after which the initial compare is stopped and records sent to Confirm Out-of-Sync phase. This parameter is valid only if you do not select the **Perform Confirm Out-of-Sync Step** parameter under the **Confirm Out-of-Sync** category.

     > **Note:**
     >
     > If **Terminate when Maximum COOS records out of sync** number is not divisible by the batch size, then there are extra records in the report. The number of records in the report is always multiple of the batch size. For example, if the batch size is 10 and **Terminate when Max COOS out of sync records** is 15, and then the report will have 20 records.

   - **Update Report file every (records)**: Enter the batch size after which the report file should be updated.
   - **Update Report file every (seconds)**: Enter the time interval after which the report file should be updated.

     > **Note:**
     >
     > **Updating Report file every (seconds)** takes precedence over **Updating Report file every (records)**, when you set values for both the parameters.

   - **Limit Number of Input Rows**: Limits the number of rows to process from the table. Useful for estimating the amount of time that a comparison will take. Limit the number of rows, then run a comparison for them. Afterward use the results to calculate how long the full comparison would take. This parameter is also useful for testing and debugging so that the run completes in a shorter time than if the entire table was processed.
   - **Source Oracle Optimizer Hint**: (Oracle) Specifies an Oracle hint that is passed to the agent on the source database when a comparison starts.
   - **Target Oracle Optimizer Hint**: (Oracle) Specifies an Oracle hint that is passed to the agent on the target database when a comparison starts.
   - **Max Concurrent Comparison Threads**: Set the number of compare pairs to be run in parallel. The remaining compare pairs is set to waiting state.
   - **Event Reporting**
     - **Generate Messages**: Select the type of message to be generated in the report. Default is None.
     - **Generate warning messages for Out-of-Sync rows after (differences)**: Set the number of out-of-sync records after which warning messages should be generated.
   - **Agent** (This is applicable only for C-Agents)

  - **Use Static Listening Port for Agent during Row Hash on Source (0 to use dynamic port list)**: Use Static Listening Port for Agent during Row Hash on Source (0 to use dynamic port list)
  - **Use Static Listening Port for Agent during Row Hash on Target (0 to use dynamic port list)**: Use Static Listening Port for Agent during Row Hash on Target (0 to use dynamic port list)

  These settings are for debugging. At the default of 0, the Oracle GoldenGate Veridata Agent is started by the Oracle GoldenGate Veridata Manager during the initial comparison step. By specifying a port number instead, you can direct Oracle GoldenGate Veridata Server to interact with Veridata Agent through that port.

- **Non Stop Process**
  - **Source Process CPU Number**: : This field designate a CPU for the process on the source system. The valid range is -1 (the default) to 16. When set to -1, the Oracle GoldenGate Manager process will attempt to start the processes in a round-robin fashion.
  - **Source Process Priority**: This field designate the priority at which the process will run on the source system.
  - **Source process name starting with = Must start with dollar symbol, followed by two letters, and end with ***: This value designate the name to use for the process on the source system. This name must start with a $ sign followed by two letters and end with an asterisk, for example $AA*.
  - **Target Process CPU Number**: This field designate a CPU for the process on the target system. The valid range is -1 (the default) to 16. When set to -1, the Oracle GoldenGate Manager process will attempt to start the processes in a round-robin fashion.
  - **Target Process Priority**: This field designate the priority at which the process will run on the target system.
  - **Target process name starting with = Must start with dollar symbol followed by two letters, and end with ***: This value designate the name to use for the process on the target system. This name must start with a $ sign followed by two letters and end with an asterisk, for example $AA*.

4. **Confirm-Out-Of-Sync**: Controls parameters for the process that performs the confirmation step.
   - **Output Out-of-Sync Record Details to Report File**: Enable this option to get the Out-of-Sync records in the compare-pair report file.
   - **Terminate when Maximum records Out-of-Sync**: Set the number of out-of-sync records, after which veridata should stop the (compare) job.
   - **Update Report file every (records)**: Enter the batch size after which the report file should be updated.
   - **Update Report file every (seconds)**: Enter the time interval after which the report file should be updated.

     > **Note:**
     >
     > **Updating Report file every (seconds)** takes precedence over **Updating Report file every (records)**, when you set values for both the parameters.

   - **Source Oracle Optimizer Hint**: Enter the source Oracle optimizer hint.

- **Target Oracle Optimizer Hint**: Enter the target Oracle optimizer hint.
- **Run Concurrently With Initial Compare**: Enable this parameter to run the Confirm Out-of-Sync and Initial Compare phase concurrently.
- **Delay Confirm-Out-of-Sync By (seconds)**: Enter the time interval by which you want to delay the Initial Compare and Confirm Out-of-Sync phases.
- **Confirm-Out-of-Sync Batch Size**: Set the batch size to be processed in the Confirm Out-of-Sync phase.
- **Perform Confirm Out-of-Sync Step**: Enable this parameter to perform the Confirm Out-of-Sync step. By default it is enabled.

5. **Repair**: Controls parameters for the repair process.
   - **Repair**
     - **Repair Batch Size**: Enter the batch size of the records to be repaired.
     - **Repair Transaction Size**: Specifies the number of repair operations included in the database transaction at the target. The special value "0" indicates that the transaction size should be the same as the repair batch size.
     - **Number of Concurrent Repair Operations**: Set the number of repair operations to be run concurrently.
     - **Check changed values**:
       * If you enable this parameter, and the non-key column records are updated post comparison, the changed out-of-sync records will not be repaired.
       * If you disable this parameter, and the non-key column records are updated post comparison, the changed out-of-sync records will also be repaired.
     - **Terminate when Maximum Repair Warnings**: Set the number of warnings after which the repair job should be stopped.
     - **Run Repair Automatically after Compare**: Enable this parameter to run repair automatically after comparison.
     - **Repair SQL Path**: Set the **Repair SQL**file path inside veridata server.
   - **Repair Reporting**
     - **Write Repair Success messages to Report**: Enable this parameter to write the repair success messages to the report file.
   - **Disable DB Triggers Session Based**
     - **Disable DB Triggers Session Based**: For databases that support disabling triggers (Oracle and Sybase), when you enable this setting, the Veridata Agent will issue the required commands to disable triggers while applying repair updates.

Specifying a Sorting Method

The Sorting Method settings of the Profile Information page specify whether data sorting will be performed by the database or by Oracle GoldenGate Veridata Server. Specify this method in the Sort Data Using option on the **Profile Information** page.

By default, Oracle GoldenGate Veridata uses the database to sort data for comparison. This default is due to historical conditions that are no longer valid. Server-side sorting is the current recommended sorting method. Database sorting should only be considered when the ordering produced by the database is identical to the ordering produced by Oracle GoldenGate Veridata Server. Following is a list of the types of conditions that will produce differing sorted ordering of the rows:

- Character encoding conditions: Oracle GoldenGate Veridata compares character data as UTF-8 encoded bytes. To match server-side ordering, key columns that contain character data must contain only ASCII data or be encoded using UTF-8, and the database must use binary comparisons for character data (no comparisons that are case-insensitive or specific to a locale).
- Some datetime data types, such as Teradata `TIME`, may sort differently in the database and in Oracle GoldenGate Veridata Server.
- To make database ordering consistent with the ordering done by Oracle GoldenGate Veridata Server, the Oracle GoldenGate Veridata Agent may add `ORDER BY` clauses to the initial comparison SELECT statement that will make the database ignore indexes on the columns. An example is `TIMESTAMP` with `TIMEZONE` data, where Oracle GoldenGate Veridata orders the data by the string representation of the data rather than by the absolute time.

When Oracle GoldenGate Veridata Server performs the sort, the Oracle GoldenGate Veridata Agents return data in the natural order that is provided by each database, and then the data is sorted by two server sort processes, one to sort source rows and the other to sort target rows. Server-side sorting supports a maximum row length of 32768 bytes. This limit normally is not exceeded when the hash comparison method is used.

Specifying Temporary Storage Directory for Source and Target Data

Specifies a location on the source disk or target disk to use as temporary storage when there is not enough memory to process all of the data that is being sorted. If no locations are defined, the default is to use a directory under the Oracle GoldenGate Veridata Server home location.

Choosing locations on different physical drives might speed up comparisons in some circumstances. You can specify multiple locations for each process, separating each one with a semicolon (for example /tmp/sort1; /tmp/sort2). All locations specified must already exist. The drives used should have sufficient free disk space. To calculate the approximate amount of space needed, use this formula:

1.5 * (Trows * (Tkey + 20)) * nTables

where:

Trows = the number of rows in table

Tkey = the average size of the table key, in bytes

nTables = the number of tables that are being compared

5.3.2 Using the Default Profile

The default Oracle GoldenGate Veridata profile is selected when a custom profile is not linked to a job or compare pair.

The name of the default profile is `$default`.

You can view and edit the settings for this default profile on the **Profile Information** page.

To edit the default profile, you must have the Oracle GoldenGate Veridata administrator role.

To edit the default profile settings:

Click **Profiles** on the Navigation pane, to display the **Profile Information** page. Click the default profile name to display the **Profile Information** page. Make the required edits and click **Save**.

To reset to system default settings:

Click **Profiles** on the Navigation pane, to display the **Profile Information** page. Click **Reset to System Defaults** button and click **Save**.

5.4 Jobs

To run comparisons, you must run a job. The job configuration determines which compare groups are processed.

> **Note:**
>
> Before creating a job, you must create at least one compare group (see Groups and Compare Pairs) to link to the job. To use customized runtime parameter settings, you must also create at least one profile (see Profiles). Otherwise, the job will use the default profile that is supplied by Oracle GoldenGate Veridata.

You must have the Administrator or Super User role to create, edit, or delete a job.

- Overview
- Creating a Job
- Running Jobs
- Editing a Job
- Monitoring Jobs
- Estimating Comparison Time
- Using the Comparison Report
- Repairing Out-Of-Sync Jobs

5.4.1 Overview

Once you have configured groups and compare pairs into jobs, you can run those jobs. Oracle GoldenGate Veridata enables you to control which groups and compare pairs are processed during any given job run, and with which runtime parameters. Once a job is running, you have easy access to views of current and finished runs.

Related Topics

Creating a Job

Editing a Job

Monitoring Jobs

Running a Job

Repairing Out-of-Sync Jobs

Estimating Comparision Time

Using the Comparison Report

5.4.2 Creating a Job

Click **Jobs** in the navigation pane.

On the **Jobs** page that opens, click **Create** and enter the following parameters:

- **Name**: Enter a job name. This is a required field and you cannot edit the name once the job is created and saved.
- **Description**: Enter a relevant description for the job.
- **Profile**: Select the profile that is associated with this job, from the drop-down list. If you do not select a profile, the default profile will be used.
- Click **Add Group**, to add a group to this job.
- On the **Add Group** page that comes up, a list of available groups is displayed.
- Select the groups that you want to add the job, and click **Add & Save**.

> **Note:**
>
> A job cannot be created without any groups added to it. You should add at least one group to a job while creating it.

- Click **Submit**.

5.4.3 Running Jobs

On the **Run Jobs** page, you can run any job that has been previously configured. Before running a job, you may want to estimate its duration. See Estimating Comparison Time .

1. Click **Run Jobs** in the navigation pane.
2. Select a job from the drop-down.
3. Select a profile from the drop-down.
4. Select the **Run Repair Automatically after Compare**, if you want to run repair automatically after the compare job is completed.
5. Click **Expand All**, to expand the Compare Pairs list and to reset any parameters.
6. Click **Run Job**.

To reset the Compare Pair Parameters:
Click **Expand All**, to expand the Compare Pairs list.

The following information is displayed in the Compare Pairs list. This information can help you to refine the job configuration.

- **Row Partitions**: Enables you to configure or override manual row partitions.
- **Compare Pair Name**: The name of the compare pair. To view actual object names, hover the mouse over the name.
- **Delta Base Time**: Shows the end time of the previous comparison, which provides a start time on which to base delta processing for the next run.

> **Note:**
>
> You have to enable delta processing option in the **Groups and Compare Pairs** page.

1. To override the selected Delta Base Time, click **Delta Base Time**.
2. On the **Delta Base Time Selection** page that opens, select the required Delta Base Time.
3. Click **Save**.

- **Source Table**: The source table selected while generating the compare pair.
- **Target Table**: The target table selected while generating the compare pair.
- Overriding Manual Row Partitions

5.4.3.1 Overriding Manual Row Partitions

1. To override a row partition, click **Override** in the **Row Partition** column.
2. In the Override page that comes up, select the partition you want to apply, or toggle the **Enable Source Partition** and **Enable Target Partition** buttons, to compare the full tables.
3. Click **Save**.

5.4.4 Editing a Job

1. From the list of jobs, click the job that you want to edit.
2. On the **Job Details** page, you can edit the description, profile, add or delete groups linked to the job.

> **Note:**
>
> You cannot edit the job name.

To add or delete Groups:

1. Click **Expand All**, in the **Linked Groups** section, to display the list of groups that are linked to this job.

   To add a group to this job, click **Add Group**.

   a. On the **Add Group** page that comes up, a list of available groups is displayed.

   b. Select the groups that you want to add to the job, and click **Add & Save**.
2. To delete a linked group, click the delete (trash can) icon next to the group.
3. Click **Save** to save the changes to the job.

5.4.5 Monitoring Jobs

- Viewing Completed Jobs
- Viewing Jobs that are Running
- Viewing Details of the Repair Jobs

5.4.5.1 Viewing Completed Jobs

To view the near-real-time information about completed jobs, click **Monitor Jobs** in the navigation pane, and select the **Completed Jobs** tab.

This page displays details of all the completed jobs:

- **Comparison Status**: The status of the comparison.
- **Repair Status**: The repair status of the comparison. It is Successful, Failed, or left blank when the repair job is not run.
- **Name**: The name of the job.
- **Start Time**: The job process start time.
- **End Time**: The job process end time.
- **Run Duration**: Time taken to process the job.
- **Compare Pair Total**: The total number of compare pairs in the job.
- **Out-of-Sync Compare Pairs**: the number of compare pairs that are out of sync.
- **Failed Compare Pairs**: The number of compare pairs failed in this comparison.
- **Cancelled Compare Pairs** : The number of compare pairs cancelled.
- **Report**: Click this link to view the job report.
- **Actions**: You can delete the Run information of the job.

To delete mutiple jobs, select the jobs from the list and click **Delete**.

By default, this page shows all the jobs that are processed. Use the Time Range Filter to constrain the view as needed.

To use the Time Range Filter

- **Most Recent Comparison Run**: Selects the most recent run for each job that exists.
- **View Last**: Selects all of the runs of all of the jobs that started within the last <n> Hours or Minutes.
- **From**: Selects all of the runs of all of the jobs that finished within a specific date range, with the option to include the time of day. Supply date and time values.

Click **Apply** to activate your selection.

Basic filter options

You can filter by:

- **Comparison Status**: show all jobs (default) or constrain the list based on jobs with a specific comparison status.

  Possible comparison status conditions:

  - In Sync
  - Out-of-Sync
  - Failed
  - Skipped
  - Cancelled

- **Repair Status**: show all jobs (default) or constrain the list based on jobs with a specific repair status.

  Possible repair status conditions:

  - Pending
  - Running
  - Out-of-Sync
  - Successful
  - Failed
  - Warning
  - Cancelled

More Details

Click an invidual job name in the list to view:

- **Job Summary**: Displays the overall job summary, **Compare Pairs Run Summary** chart, and the list of compare pairs. Click a compare pair name to view the **Compare Pair Details**, **Performance**, and the **Report** tabs.
  - **Compare Pair Details** section:
    - Displays a chart view of the comparion details and a graph view of the out-of-sync operations.
    - Displays the Out-of-Sync and Skipped Row Details, which you can filter by Status, Operation, or Columns.
    - You have an option to download Repair SQL or to repair the compare pair.

      > **Note:**
      >
      > Download Repair SQL option is available only for Jobs that have been configured for datasource connections with the Oracle datatype.

  - **Performance**: Displays the comparison performance details at different comparison stages such as Sorting, Initial Compare, and Confirm Out-of-sync.
    The page also displays a graph view with the performance details for each stage, which you can filter based on:
    - Run Duration
    - Rows Processed
    - Rows Per Second
  - **Report**: Displays the following details about the selected comparison:
    - **Overiew**
    - **Profile**
    - **Parameters**
    - **Agent Info**
    - **Performance Stats**
    - **Sort Statistics**

    - **Summary**

      You can download the individual comparison report for a selected compare pair, from this page.

  - From the Job Summary page, click **More Actions** for the **Repair** and **Download Repair SQL** options.
  - Click **Run Job**, to run the current job again.
- **Out-of-Sync Summary**: Displays the out-of-sync details at each group and compare pair levels. You can filter the details by:
  - Groups
  - Status
  - Operation
  - Columns

  You can also download the Repair SQL or repair the target from this page.
- **Report**: Displays the the report for a specific job, group, or compare pair. The report shows processing results and the causes of errors:
  - **Results**
  - **Overiew**
  - **Profile**
  - **Run Options**
  - **Summary**

  You can download the following reports from this page:
  - Overall Job Report
  - Job and Compare Pairs Reports

5.4.5.2 Viewing Jobs that are Running

To view the near-real-time information about running jobs, click **Monitor Jobs** in the navigation pane, and select the **Running Jobs** tab.

The list view shows statistics and status for the running comparisons. By default, the list is organized within the context of the job as a unit. You can filter the list to organize itby time range, or by comparison status.

- **Comparison Status**: The current run status of the comparison. For example, Running, Waiting.
- **Name**: The name of the job.
- **Start Time**: The job process start time.
- **Run Duration**: The duration for which the job has been running.
- **Out-of-Sync Compare Pairs**: The number of out-of-sync compare pairs.
- **Waiting Compare Pairs**: The number of compare pairs which are yet to be processed.
- **Running Compare Pairs**: The number of compare pairs that are currently being processed.
- **Failed Compare Pairs**: The number of compare pairs that have failed the comparison job.

- **Cancelled Compare Pairs**: The number of cancelled compare pairs.
- **Processed Compare Pairs**: The number of all the processed compare pairs. This includes in-sync, out-of-sync, and failed compares pairs.
- **Actions**: Click the stop (close) icon to the stop the job.

By default, this page shows all the jobs that are processed. Use the Time Range Filter to constrain the view as needed.

To use the Time Range Filter

- **Most Recent Comparison Run**: Selects the most recent run for each job that exists.
- **View Last**: Selects all of the runs of all of the jobs that started within the last <n> Hours or Minutes.
- **From**: Selects all of the runs of all of the jobs that finished within a specific date range, with the option to include the time of day. Supply date and time values.

Click **Apply** to activate your selection.

Basic filter options

You can filter by:

- **Comparison Status**: show all jobs (default) or constrain the list based on jobs with a specific comparison status.

  Possible comparison status conditions:

  - In Sync
  - Out-of-Sync
  - Failed
  - Skipped
  - Cancelled
  - Running
  - Cancelling

More Details

Click an invidual job name in the list to view:

- **Running Job** : Displays the following details.
  - Start Time
  - Run Duration
  - Comparison Status
  - Option to stop the running job
  - The **Compare Pair Status** section dispays the graph view of compare pairs in each stage.
  - Click an individual **Compare Pair Name** in the list, to view the **Compare Pair Details** page:
    * You can view the individual compare pair details and a graph view of total out-of-sync rows in the **Intial Compare**, **Persistent Out-Of-Sync** stages, and the rows **In-Sync After In Flights**.

* You can stop an indivual compare pair comparison from this page.

5.4.5.3 Viewing Details of the Repair Jobs

To view the information about the repaired jobs, click **Monitor Jobs** in the navigation pane, and select the **Repair Jobs** tab.

- **Repair Status**: The status of the repair. For example, Successful, Failed.
- **Name**: The name of the job.
- **Start Time**: The repair job process start time.
- **Run Duration**: Time taken to process the repair job.
- **Processed Compare Pairs**: The total number of compare pairs in the repair job.
- **Repaired Compare Pairs**: The number of compare pairs which were successfully repaired.
- **Failed Compare Pairs**: The number of compare pairs that have failed the repair.
- **Cancelled Compare Pairs**: The number of cancelled compare pairs.
- **Actions**: Click the stop (close) icon to the stop the job.

To use the Time Range Filter

- **Most Recent Comparison Run**: Selects the most recent run for each job that exists.
- **View Last**: Selects all of the runs of all of the jobs that started within the last <n> Hours or Minutes.
- **From**: Selects all of the runs of all of the jobs that finished within a specific date range, with the option to include the time of day. Supply date and time values.

Click **Apply** .

Basic filter options

You can filter by

- **Repair Status**: show all jobs (default) or constrain the list based on jobs with a specific comparison status.
  - Pending
  - Running
  - Out-of-Sync
  - Successful
  - Failed
  - Warning
  - Cancelled

More Details

Click an invidual repair job name in the list to view:

- **Repair Summary** : Displays the list of all the Compare Pair in the repair job.
  - Click an individual compare pair name in the list, to view the **Repair Details** page.
  - You can filter the Repair Row Details by:

    - Status:
        - Pending
        - Running
        - Out-of-Sync
        - Successful
        - Failed
        - Warning
        - Cancelled
    - Operation:
        - Insert
        - Update
        - Delete
    - Columns:
        - Key
        - Out-of-Sync

- **Report**: Displays the the report for a specific job, group, or compare pair. The report shows processing results and the causes of errors:
    - **Results**
    - **Overiew**
    - **Profile**
    - **Run Options**
    - **Summary**

    You can download the following reports from this page:
    - Overall Repair Job Report
    - Job and Compare Pairs Repair Reports

5.4.6 Estimating Comparison Time

When tables are large, you might want to estimate the amount of time that a comparison will take before running a full comparison. To get an estimate, run a test comparison of a limited number of rows. For example, if there are 100 million rows in a table, you can run a comparison for the first million rows and then multiply that amount of time by 100.

To specify the number of rows compared

1. On the navigation pane, click **Profiles**.
2. In the list, click the name of the profile that is linked to the job (or select it, and then click edit (pencil) icon.
3. In the **Profile Information** section, Click **Initial Compare**.
4. Clear the **Use Default** toggle of the **Limit Number of Input Rows** parameter, and then type the number of rows that you want to compare.
5. After running the test comparison, change **Limit Number of Input Rows** back to the default for the full comparison run.

5.4.7 Using the Comparison Report

To view a comparison report, use the Reports page.

A comparison report is generated for each job, group, and compare pair that is finished being processed. It contains summary details about out-of-sync row counts, the number of records processed, performance statistics, errors, and so forth.

The comparison report tells you how extensive an out-of-sync problem is. It also provides performance statistics and, optionally, column details. The comparison report can be viewed by any user role.

To view a report

To view a report, click the name of the report in the Report Name column of the Existing Reports list.

Alternately, you can click the **Select** button in the list next to the name of the report, and then click **View**. The Report View page will be displayed.

Viewing the report file on disk

The report is a text file that is stored on disk. It can be viewed from its location on disk.

5.4.8 Repairing Out-Of-Sync Jobs

To repair Out-of-Sync Comparisons

Click Monitor Jobs in the navigation pane, and click the Completed Jobs Tab.

Click a job name in the list to view the Job Summary.

- Click **More Actions** and click **Repair**.
- To repair an individual compare pair, click the compare pair name in the list to open the **Compare Pair Details** page, and click **Repair**.

Oracle GoldenGate Veridata allows repair of duplicate records for the tables that do not have Primary Key's in the following databases: Oracle, MySQL and MSSQL. The Details for Out-Of-Sync page is displayed.

> **Note:**
>
> If the table contains duplicate records, then it is suggested to use System Generated Mapping in Compare-Pair. If you decide to use User-defined mapping then, ensure that the user-defined column mappings have unique dataset. If the row has Skipped status icon, then the row is ignored from Repair.

The Repair Jobs page displays a summary of all repair jobs. You can use the Filters on this page to display older repair jobs and to filter repair jobs by repair status and job name. Following are the possible repair statuses:

- Pending
- Running
- Out-of-Sync

- Successful
- Failed
- Warning
- Cancelled
- Downloading SQL Statements for Out-of-Sync Records
  Oracle GoldenGate Veridata provides the Download Repair SQL functionality to view the SQL Queries generated as part of Veridata Repair. The Download SQL Queries is enabled only when either Oracle or SQL Server is used as the target database.

5.4.8.1 Downloading SQL Statements for Out-of-Sync Records

Oracle GoldenGate Veridata provides the Download Repair SQL functionality to view the SQL Queries generated as part of Veridata Repair. The Download SQL Queries is enabled only when either Oracle or SQL Server is used as the target database.

You can look at these SQL Statements before the Oracle GoldenGate Veridata executes them onto the target database, or execute these SQL statements by yourself on any of the other database tools. With the Download Repair SQL functionality, you can download the SQL statements for all your out-of-sync-records, and can also execute them at your convenience. To download SQL queries:

1. From the **Finished Jobs** page, select the out-of-sync comparisons.

   You can select Jobs, groups, and compare pairs for downloading SQL statements.

2. To download SQL queries for out-of-sync data, select the job from the finished jobs and click **Download Repair SQL**.

   The **Download Repair SQL** button is enabled only for users who have access to the repair access functionality.

   The SQL files are downloaded to the browser where the application is being used. When you click **Download Repair SQL**, the file gets downloaded to the browser.

If the record is already repaired, then the check box against the record is replaced with date and time of repair. Those records are not available for downloading SQL statements.

- Executing the SQL File

5.4.8.1.1 Executing the SQL File

To execute the SQL file:

1. When you click **Download Repair SQL**, a SQL file is generated. Copy this zip to the target system. In order to enable access to target Oracle DB, run the following query on the target database. This is a one-time operation only:

   ```
   CREATE OR REPLACE DIRECTORY VDT_LOB_DIR AS '<path of lob files>';
   ```

   For example:

   ```
   CREATE OR REPLACE DIRECTORY VDT_LOB_DIR AS '/scratch/lobs';.
   ```

   Here:

   `/scratch/lobs` is the path on the target system where the generated SQL files and LOB files should be copied. When the SQL scripts are executed from the DB prompt, the LOB files are read from `/scratch/lobs`

2. Unzip the zip file.

LOB files are generated only when the table has BLOB datatypes and huge data for CLOB/NCLOB.

The directories are created as follows:
`<JobName><Timestamp>`

- *<GroupName>*
  - SQL file (format – <ComparePairName>.sql)
  - LOB file (format - <TableName>_<ColumnName>_<Timestamp>.lob)
- *<Group2>*
  - SQL file
  - LOB files

o SQL file (format – <ComparePairName>.sql)

o LOB file (format - <TableName>_<ColumnName>_<Timestamp>.lob)

3. Copy the SQL and LOB files to `VDT_LOB_DIR`. However, it is not mandatory to copy the sql file to `VDT_LOB_DIR`. But, the LOB files are required in `VDT_LOB_DIR`.
4. If the data has any multibyte characters, then ensure that the `NLS_LANG` environment variable is set before launching the DB Terminal.

   On Linux:

   a. Bourne/Bash Shell
      - `export NLS_LANG=<NLS_LANGUAGE>_<NLS_TERRITORY>.<NLS_CHARACTERSET>`. For example, `export NLS_LANG=AMERICAN_AMERICA.AL32UTF8`

   b. C/TCSH Shell
      - `setenv NLS_LANG <NLS_LANGUAGE>_<NLS_TERRITORY>.<NLS_CHARACTERSET>`. For example, `setenv NLS_LANG AMERICAN_AMERICA.AL32UTF8`

   On Windows:

   - Set the NLS_LANG in System Variable. Value should be `<NLS_LANGUAGE>_<NLS_TERRITORY>.<NLS_CHARACTERSET>`. For example: `AMERICAN_AMERICA.AL32UTF8`

   Note: Launch the DB terminal after `NLS_LANG` is set.
5. Login to DB terminal and execute the SQL file: `SQL> @<sql file>`. For example: `SQL> @VeridataRepair.sql`. If prompted, enter the value of `NLS_CHARACTERSET`.

> **Note:**
>
> If the SQL file is in a different location than DB terminal is open, then ensure to include the complete path of the SQL file while executing it. For example, SQL> `@/Home/User/filename.sql`.

5.5 Users

You can manage users and user groups from the **User Management** page. You can create, delete, edit, users and user group.

- Configuring User Groups

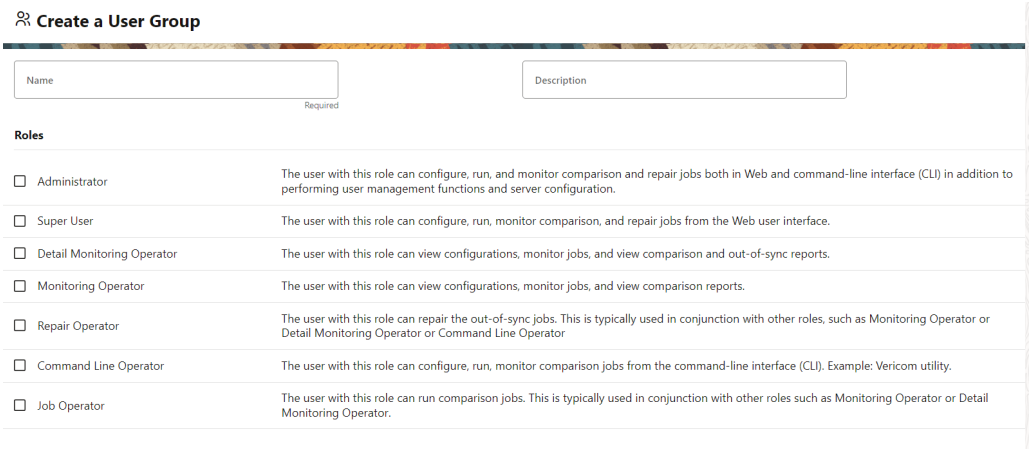
- Configuring Users

5.5.1 Configuring User Groups

> **Note:**
>
> - You need to have an Admin User role to access this page.
> - You can select from the exisiting default user groups.

To create a User Group:

1. Click **User Management** from the Navigation pane. On the **User Management** page, select the **User Groups** tab and click **Create**, to display the **Create a User Group** page.
2. Enter the following parameters and click **Next**:
   - Select a user role listed under the **Roles** tab, and click **Submit**.



> **Note:**
>
> - Configurations available to create, edit, view, delete:
>   * Connections
>   * Groups and Compare Pairs
>   * Jobs
>   * Users
>   * Utilities
> - Job Operator and Repair Operator role is used in conjunction with other roles such as the Monitoring Operator or Detail Monitoring Operator. You cannot login to the UI, having only the Job Operator or the Repair Operator role.

To view User Group details

1. Select a user group from the list under the **User Groups** tab.
   You can view the user group details such as the Roles and Role descriptions under the **User Group Details** tab. You can view the users associated with the selected user group under the **Users** tab.

To update a User Group

1. Select a user group from the list under the **User Groups** tab.
   You can edit the description and roles under the **User Group Details** tab.

   You can edit the users associated with the selected user group under the **Users** tab.
2. Click **Save**.

To delete a User Group

1. Under the **User Groups** tab, click the 3 dots (…) next to the user group that you want to delete.
2. Click **Delete**.
3. Click **Delete** on the confirmation screen.

5.5.2 Configuring Users

Users with Adminstrator role can access this page.

To create a User:

1. Click **User Management** menu option in the Navigation pane to display the **User Management** page.
2. Select the **Users** tab and click **Create**.
3. On the **User Details** page, enter the following parameters and click **Next**:
   - **Name**: Enter an username. You cannot edit the name once created.
   - **Description**: A description is optional and can be edited as needed.
   - **Email ID**:Enter an email ID.
   - **Password**: Enter a password.
   - **Confirm Password**: Retype the password to confirm it.
4. On the **User Group** screen, all the available user groups are listed. Select and add all the user groups that you want to add the user to, and click **Submit**.

To view User

1. Click **User Management** menu option in the Navigation pane to display the **User Management** page.
2. Select a user from the list under the **Users** tab.
   You can view the user and the user group details on the right side of the screen, under the **User Details** and the **User Groups** tabs.

To update User details:
You can update the user and the user group details on the right side of the screen.

- **User Details**: You can edit the description and reset the password. After editing, click outside the text fields, to enable the **Save** option, and click **Save**. You cannot edit the user name, once created.
- **User Groups**: You can add or remove User Groups associated with the seleted user, and click **Save**.

To delete a User:

1. Select a user from the list under the **Users** tab. Click the 3 dots (…) next to the user name and select **Delete**.
2. Click **Delete** on the confirmation screen.

5.6 Utilities

You can use the Oracle GoldenGate Veridata Import and Export utilities from the **Utilities** menu of the Veridata Web user interface. For more information, see Running the Export and Import Utilities .

- Export
- Import

5.6.1 Export

To export Jobs:

1. Click **Utilities** from the menu to display **Utilities** page. The **Export** tab is displayed by default.
2. Select jobs from the **Available Jobs** list and move them to **Jobs to export**.
3. Click **Export** to export these jobs.

The selected Jobs are exported to `export.xml`.

For more information, see Running the Export and Import Utilities

5.6.2 Import

To import Jobs:

1. Click **Utilities** from the menu to display **Utilities** page. The **Export** tab is displayed by default.
2. Click **Import**. You can select either an exported file (`.xml`) or a GoldenGate parameter file (`.prm`)
3. Click **Browse** File, navigate to the location of the file, select the file, and then click **Import**.

A successful import message is displayed after the file import is complete.

6
Administer

6.1 Managing C-Agent and Java Components

This topic describes how to manage C-agent and Java-based components, including controlling logging levels and connecting to the web user interface.

This topic includes the following sections:

6.1.1 Starting and Stopping the Veridata Server and the Repository

To start the Oracle GoldenGate Veridata server:

1. Go to `<VERIDATA_HOME>/bin`.
2. Run the following command `./run.sh` (starts the repository also in case it is down).
   Or

   `./run.sh skipRepoStart` (if you want to ignore the repository status and just start the server)

To stop the Oracle GoldenGate Veridata server:

1. Go to `<VERIDATA_HOME>/bin`.
2. Run the following command `./run.sh stop`.

To start and stop the repository:

1. Go to `<VERIDATA_HOME>/bin`.

2. Run the following commands:
   - To start the repository: `./run.sh repoStart`.
   - To stop the repository: `./run.sh repoStop`.
   - To know the repository status: `./run.sh repoStatus`.

6.1.2 Starting and Stopping the C-Agent

When the Oracle GoldenGate Veridata (server) initiates comparisons, the C-agent starts automatically. However, for Oracle GoldenGate Veridata Agent (agent) to function correctly, the following must be running:

- The database to which the agent is linked.
- The Manager process for the C-agent.

Although the agent process itself is automatic, you can stop the Manager process that controls the agent. Stopping Manager prevents the server from being able to start a new agent process, but it does not stop agents that are already running.

To control the C-agent Manager:

1. From the agent installation location, run the Oracle GoldenGate software command-line interface.
2. Stop or start the Manager.

```
START MANAGER
STOP MANAGER
```

6.1.3 Starting and Stopping the Java-Based Components

The server and web user interface components are Java programs. The agent component is also available as a Java program for all platforms except NonStop.

> **Note:**
>
> Before starting the server and web processes, start the repository database.

To start and stop the Java agent component:

1. Navigate to the `AGENT_DEPLOY_HOME` directory.
2. Start or stop the agent.

   UNIX or LINUX
   `agent.sh {start | run}` OR `agent.sh stop`

   - In these commands:
   - `run` starts the agent in the same command window where it is launched.
   - `start` starts the agent in a separate command window.

> **Note:**
>
> The `run` option is useful for diagnosing errors that happen during the startup process before the agent error logging is configured. When the `run` option is used, messages written to `stdout` and `stderr` appear in the command window. The agent normally logs its messages to the log file, so only operating system messages and logging system errors are written to `stderr`. When the start option is used, messages written to `stdout` and `stderr` are discarded.

Configure the host to start and stop the processes automatically. Contact your system administrator if you need assistance.

6.1.4 Reloading Logging Information

You can reload logging information from the `<VERIDATA_AGENT_HOME>/config/odl.xml` configuration file to a running agent.

To reload logging information, start the agent and run the UNIX/Linuz `reloadLog` command as follows: `agent.sh reloadLog`

6.1.5 Connecting to the Oracle GoldenGate Veridata Web Interface

To connect to the Oracle GoldenGate Veridata Web user interface:

1. In a Web browser, enter the following address: `http://hostname:port/veridata`.

   In this example, `hostname` is the name of the system where the server is installed and `port` is the port number where it is running (default is 8831). Use `localhost` as the host name if you are connecting on the system that is local to the server installation.

2. On the Oracle GoldenGate Veridata Web login page, enter your user name and password. For full instructions on using the Oracle GoldenGate Veridata web user interface, see the online help.

6.2 Vericom

The `Vericom` command-line interface provides a tool for you to manage identities in the credential store and run comparisons.

This topic includes the following sections:

- Overview of the Vericom Tool
- Supported Commands in Vericom
- Vericom Output Examples

6.2.1 Overview of the Vericom Tool

You can use the `vericom` tool to execute certain comparison tasks from the command shell of the operating system. The `vericom` tool runs the Oracle GoldenGate Veridata command-line interface and enables you to handle these activities with automated programs.

You can:

- Run an entire job or a specific group within a job or a specific compare pair of a group and job.

For specific compare pairs, you can:

- Review previous out-of-sync results
- Override the same profile and row partition settings that are possible from the web user interface

You can also run comparisons from the Oracle GoldenGate Veridata web user interface. This interface provides greater control for configuring the objects to be compared and for controlling runtime parameter settings.

> **Note:**
>
> If the **SSL / TLS Security for Web** option is selected during the Oracle GoldenGate Veridata installation, then do the following for Vericom to work:
>
> - Copy the `<VERIDATA_HOME>/config/vdtWebKeystore.p12` to `<VERIDATA_HOME>/cli/config/veridata-23c.p12`

6.2.2 Supported Commands in Vericom

The following commands are supported in Oracle GoldenGate Veridata:

1. **-u OR -user :** Veridata user running veridata job
2. **-host**: Veridata helidon hostname, bydefault , it is localhost
3. **-port :** Veridata Server port. Optional paramter, if not passed then default port will be taken
4. **-j OR -job <jobname> :** Veridata job name to be executed
5. **-j OR -job <jobname> -g OR -group <groupName> :** If any specific group within a job to be executed
6. **-j OR -job <jobname> -g OR -group <groupName> -c OR compare_pair <CPName>:** If any specific Compare Pair needs to be executed
7. **-repair :** If repair has to be run along with Compare
8. **-p OR -profile <profileName> :** To override Job profile at runTime.
9. **-pS :** Source Partition Name (It takes default partition name)
10. **-pS <source partition name> :** To override source manual row partition at runTime
11. **-pT:** Target partition Name (It takes default partition name)
12. **-pT <target partition name> :** To override target manual row partition at runTime
13. **-delta :** Run with delta enabled. When not passed, all rows will be compared in every run
14. **-help:**Parameter to provide Vericom usage
15. **-v :** Should provide Veridata Server version details.
16. **-monitorStats** `<runId>` : This command is to monitor the Job status. It takes the `runId` as input.

Examples

1. When running the Job via Vericom, it is necessary to show the job status as follows: `./vericom.sh -j <jobName>`.

   Output:

   ```
   Run ID: (2258, 0, 0)
   ```

2. To fetch the Run stats, run the command as follows: `./vericom.sh -monitorStats <runId>`:

   Output:

   ```
   Run ID: (2257, 0, 0) Job Start Time: 2008-03-21 22:48:05 Job Stop Time: 2008-03-21
   22:48:20  Number of Compare Pairs: 3 Number of Compare Pairs With Errors: 0 Number
   of Compare Pairs With OOS: 1 Number of Compare Pairs With No OOS: 1 Number of
   Compare Pairs Cancelled: 0 Job Completion Status: SUCCESSFUL
   RUN ID: <CP RUN ID> <CP Name> <Group Name>
   RUN ID: <CP RUN ID> <CP Name> <Group Name>
   RUN ID: <CP RUN ID> <CP Name> <Group Name> and so on.
   ```

3. To View Compare Pair details, run `./vericom.sh -monitorStats <comparepair_runId>`
   Output

   ```
   Run ID: (2257, 0, 0) Job Start Time: 2008-03-21 22:48:05 Job Stop Time: 2008-03-21
   22:48:20  Number of Compare Pairs: 3 Number of Compare Pairs With Errors: 0 Number
   of Compare Pairs With OOS: 1 Number of Compare Pairs With No OOS: 1 Number of
   Compare Pairs Cancelled: 0 Job Completion Status: SUCCESSFUL
   Compare Pair RUN ID:
   Compare Pair Name:
   Compare Pair ID:
   Phase:
   Status:
   Total Rows Compared:
   Rows In-sync:
   Rows out-of-sync:
   Compare-pair report:
   ```

> **Note:**
>
> The Vericom utility fails when a special character, such as the $ symbol, is present in the name fields of Job, Group, or Compare Pairs. To resolve this issue, an escape character needs to be added, as follows:
>
> **Actual Compare Pair Name**:
> `\THOR.$DATA08.IRFASRC.EACCT=\THOR.$DATA08.IRFASRC.BACCT`
>
> **With Escape Character**:
> `\THOR.\$DATA08.IRFASRC.EACCT=\THOR.\$DATA08.IRFASRC.BACCT`
>
> Run the command as follows:
>
> ```
> ./vericom.sh -wluser veridata -j NskTest -g nskENscribeTesting -c
> \THOR.\$DATA08.IRFASRC.EACCT=\THOR.\$DATA08.IRFASRC.BACCT
> ```

6.2.3 Vericom Output Examples

To view the results of a comparison that you run with the vericom tool, you can use the Oracle GoldenGate Veridata web user interface to view the comparison report. You can also view the

output that is returned by the tool to the terminal. If a run finishes successfully, statistics for the job are displayed.

The following examples use the `TestJob` job:

Example 1
The example shows a run on a Linux system. The process exits with status 0, and finished job statistics are not displayed.

```
./vericom.sh -user veridata -port 8089 -job jobBetaBug
JAVA_HOME is set to: /Library/Java/JavaVirtualMachines/jdk-17.jdk/Contents/Home
Java version:17.0.8
[Enter user password:]
Veridata job submitted successfully!
Run ID: 325865/0/0
Execute ./vericom.sh -user <username> -monitorStats <runId> to fetch the run statistics.
```

Example 2
This example shows how to get statistics from the Run Id of a Job.

```
/vericom.sh -user veridata -monitorStats 325865/0/0
JAVA_HOME is set to: /Library/Java/JavaVirtualMachines/jdk-17.jdk/Contents/Home
Java version:17.0.8
[Enter user password:]
Run ID: 325865,0,0
Job Start Time: 2024-06-19T05:32:42.870Z
Job End Time: 2024-06-19T05:32:42.990Z
Number of Compare Pairs: 1
Number of Compare Pairs With Errors: 1
Number of Compare Pairs With Out-Of-Sync: 0
Number of Compare Pairs With No Out-Of-Sync: 0
Number of Compare Pairs Cancelled: 0
Number of Compare Pairs Running: 0
Job Completion Status: ERRORS
Compare Pair Run ID: 325865,0,0
Compare Pair Name: ETMSP_TIMESTAMPTEST=ETMSP_TIMESTAMPTEST
Compare Pair Id: 1,893
Compare Pair Phase: FINISHED
Compare Pair Status: FAILED
Number of Rows Compared: 0
Number of Rows In Sync: 0
Number of Rows Out Of Sync: 0
```

6.3 Veridata Export and Import Utilities

You can use the Export and Import utilities, provided with the Oracle GoldenGate Veridata installation, to define portions of your configuration. Starting Oracle GoldenGate Veridata 23c, you can access these Utilities from the Web User Interface.

- Introduction to the Export and Import Utilities
- Running the Export and Import Utilities
- Configuration File Element Reference

6.3.1 Introduction to the Export and Import Utilities

Using the Export and Import utilities, you can create XML documents that are used to configure Oracle GoldenGate Veridata. The DTD (Document Type Definition) that governs these XML documents is stored in the *VERIDATA_23C_HOME*`/cli/vdt-export-client.jar` file.

The Export utility helps you to either selectively or completely export the compare configuration data to an XML file. Additionally, you can use it to export configurations between different Veridata repository types using the import functionality. For example, from a SQL Server configuration to an Oracle configuration.

The Import utility allows you to configure database connections, comparison groups including compare pairs, comparison jobs, and profiles. It takes an XML document as input then creates comparison objects in Oracle GoldenGate Veridata. Typically, the XML document matches the inputs on the configuration pages in the user interface.

You should have an understanding of basic XML and its rules.

These utilities provide the following advantages:

- It can reduce the time required to define repetitive tasks
- It allows you to create reusable configurations
- It can ensure that your test configuration mirrors the one you use for production
- Supported Configurations

6.3.1.1 Supported Configurations

Oracle GoldenGate Veridata import and export utilities support configuring:

- Database connections
- Comparison groups (jobs, groups, and compare pairs)
- Profiles

6.3.2 Running the Export and Import Utilities

The Export and Import utilities run from the *VERIDATA_23C_HOME*/cli/bin directory of the Oracle GoldenGate Veridata installation location. The UNIX and Linux scripts for Import and Export are vdt-import.sh and vdt-export.sh respectively.

- Using the Export Utility
- Using the Import Utility
- Processing the Configuration
- SSL Configuration for Export and Import Utilities

6.3.2.1 Using the Export Utility

The syntax for running the export utility is:

```
vdt-export [-host <host> ] [-port <port> ] -user <user Name> -export <fileName>
```

Required Parameters

- -host *<host>*: The host for Veridata server. The default is localhost.
- -port *<port>*: The port number for the Veridata Web server. The default value is 8089.
- -user *<user Name>*: The Oracle GoldenGate Veridata user. For example, *veridata*.
- -export *<fileName>*: The file where the Veridata config file gets exported to.

Optional Parameters

- One of these optional operations can be requested at run time:
  - `-help`: Prints the usage of all the Export Config file flags.
  - `-version v`: Prints version.
  - `-jobs [<job1> <job2> ... <jobN>]`: Job Names space separated for exporting Configuration.
  - `groups [<group1> <group2> ... <groupN>]`: Export all groups in the repository or add group names separated by a space, such as `group1`, `group2`, or `group3`.
  - `-connections [<conn1> <conn2> ... <connN>]`:Export all connections in the repository or add connections separated by a space, such as `conn1`, `conn2`, or `conn3`.
  - `-profiles [<profile1> <profile2> ... <profileN>]`: Export all profiles in the repository or add profiles separated by a space, such as `profile1`, `profile2`, or `profile3`..
  - `-all`: All the repository configuration will be exported, takes precedence over above options. Also this is the default behavior when none of the optional flag values are entered.
  - `-exportPassword`: Passwords for connections that are exported. By default, the Connection passwords are exported.
  - `-ssl`: To communicate with Oracle GoldenGate Veridata 23c using the `https` protocol.

6.3.2.2 Using the Import Utility

The syntax for running the Import utility is:

```
./vdt-import.sh < -create| -delete | -update | -replace > <fileName> -user <username> -
host
<hostname> -port <port> [more_flags]
                        | -p
                        | -extract
                        | -ssl
                        | -help | { -version | -v}
```

Required Parameters

You need to specify one of the following required parameters while running the Import utility:

- `-create <fileName>`: Create Oracle GoldenGate Veridata config or parameter from file `<fileName>`.
- `-delete <fileName>`: Deletes Oracle GoldenGate Veridata config from file `<fileName>`. All named items that exist for the configuration are removed from the repository.
- `-update <fileName>`: Updates Veridata config from file `<fileName>`. New items are added to the repository and existing items are modified. Items existing in the repository and not listed in the configuration are deleted.
- `-replace <fileName>`: Replaces Veridata config from file `<fileName>`. All items listed to be replaced in the configuration are replaced as specified.
- `-user <username>`: Oracle GoldenGate Veridata user.
- `-host <hostname>`: Server host where Oracle GoldenGate Veridata is running.
- `-port <port>`: Server port where Oracle GoldenGate Veridata is running.

- *<fileName>*: Veridata `config.xml` or GoldenGate parameter file path to perform create, delete, update, or replace operation .

Optional Parameters

- One of these optional operations can be requested at run time:
  - `-p`: Properties file path.
  - `-extract`: Extract file path and name.
  - `-ssl`: To communicate with the veridata 23c using https, which is the `tls` protocol.
  - `-help`: Prints usage.
  - `-version v`: Prints version.

6.3.2.3 Processing the Configuration

The import utility first parses the *configuration.xml* file attempting to complete the entire file before aborting due to the errors. Any errors it finds are logged in the following location:

DOMAIN_HOME/*<VERIDATA_23C_INSTALL_LOCATION>*/cli/veridata/logs/vdt_import.log

If it does not abort because of errors, it makes a second parsing pass, this time processing the configuration.

Matching Object Names

Database object names, such as catalogs, schema, tables, indexes, and columns will be matched according to these rules:

- The matching is case insensitive
- The hyphen (-) is considered a match to the underscore (_) to support matching Enscribe DDL and SQL columns
- Wildcard expressions for table names and source column names match against the exact name and against the uppercase version of the name.
- Quoted names for schemas and wildcards match everything within the quotations must be matched exactly. A wildcard character within quotes is treated as an ordinary character. An example of a quoted name as it would appear in the XML is:

  ```
  source-table="&quot;CHAR_TYPES&quot;*"
  ```

  This would match `CHAR_TYPES`, `CHAR_TYPES2`, and `CHAR_TYPES_NOTNULL`.
- Filters can either include or exclude schemas and tables. If include filters are used, at least one filter must be matched before a table can be included in a compare pair. If exclude filters are used, a table is excluded if it matches any exclude filter. Include filters can include a `COLFILTER` element that contains a list of columns to include or exclude. When a table matches a include filter, the include filter's `COLFILTER` is used to specify the columns for the generated compare pair. The schema and table name can use wildcards.

  Filters can be used to exclude tables with specific names. For example, you can use the following pattern to exclude tables containing FIRST in their names:

  ```
  <filter type="exclude" table="*FIRST"/>
  ```

Filters can be used to exclude views by specifying as follows in the XML:

```
<filter exclude-views="true"/>
```

.

For NonStop Enscribe files, file pattern filters are used. The file pattern is any valid NonStop file name pattern.

- A compare pair may have a column specification with the Boolean attribute "optional". When this attribute is true, the column is only included in the compare pair if the source table includes the specified source column.

Determining Key Columns

The key columns are selected in the following order:

1. Explicit key column definitions if they are available. In this case if `source-pkey` and `target-pkey` `compare-pair` element attributes are set it will generate an error.
2. Columns in the index specified by `source-pkey` and `target-pkey` attributes of the `compare-pair` element. The number of columns and all data types must match and the data types must be compatible.
3. Columns in the system-selected primary key.

Generating Compare Pairs

Compare pair generation has the following characteristics:

- Generating from wild cards works the same as the user interface generation except that regular expressions can be used.
- Compare pairs are processed in the order specified in the *`configuration.xml`* file
- The compare pairs generated by a single compare pair element are generated in alphabetical order of the source table name.
- When compare pairs are generated by more than one compare pair element, the first one will be used.

As a general rule, the order of the compare pair elements should be:

1. Compare pairs with specialized configuration requirements, such as user-defined keys.
2. Compare pairs that match general patterns.
3. Exclusions of compare pairs that would otherwise match general patterns.

6.3.2.4 SSL Configuration for Export and Import Utilities

The Export and Import utilities are now supported to communicate with Oracle GoldenGate Veridata server using SSL protocol.
To configure SSL for Export and Import Utilities:

1. Check whether the SSL ceritificate is self-signed or not in the Oracle GoldenGate Veridata 23c Server. If its a self-signed certificate, then the client certificate is shipped as part of the installer under *`<install_location>`*`/config/vdtWebKeystore.p12`.
2. Copy the *`<install location>`*`/config/vdtWebKeystore.p12` to *`<install location>`*`/cli/config/veridata-23c.p12`.

> **Note:**
>
> To enable SSL, there is no modification required for the self-signed certificate. The Oracle GoldenGate Veridata 23c server should start with SSL.

3. Access the CLI utility through the SSL mode as follows:

   ```
   ./vericom.sh -j <job> -user <user> -ssl
   ```

   Here `-ssl` is the option for ssl communication between the cli and Server.

4. If you have your own ssl certificate, replace the *<install_location>*`/cli/config/veridata-23c.p12` to the client certificate and rename it to `veridata-23c.p12`.

6.3.3 Configuration File Element Reference

The configuration is defined by the top level `configuration` element and several nested elements. Most of these elements have attributes that define their characteristics, such as the `operation` attribute for the `configuration` element or the `port` attribute for the `connection` element.

The following is the high-level element hierarchy in the configuration XML file. For more information about an element and its attributes, click the element name in the hierarchy.

configuration
- connection
  - conn-properties
- group
  - description
  - filter
  - sql-partition
  - enscribe-key
  - compare-pair
    - enscribe-info
    - enscribe-key
    - sql-partition
    - column
    - excluded-column
    - delta-config
- job
- profile

- configuration
- column
- colfilter
- colfiltercol
- compare-pair
- connection
- conn-properties
- delta-config
- description

- enscribe-info
- enscribe-key
- excluded-column
- expandddl
- filter
- group
- job
- profile
- key-column
- profile-general
- sorting-method
- initial-compare
- confirm-out-of-sync
- param
- repair
- sql-partition
- table partition
  The `table-partition` element helps in specifiying the partition name details in the Oracle GoldenGate Export and Import tools.

6.3.3.1 configuration

The root element is `configuration`.

The following elements can be nested within the `configuration` element:

Table 6-1 configuration Elements

Elements	Description
`connection`	One or more Veridata database connection definitions.
`group`	One or more Veridata comparison group definitions.
`job`	One or more comparison job definitions.
`profile`	One or more profile definitions.

The following attributes describe the `configuration` element:

Table 6-2 configuration Attributes

Attribute	Description
`validation`	Specifies the type of validation that is used for the configuration. The options are: "`required`" - All compare pairs must be successfully validated before any pairs are added to the repository. This is the *default* value. "`omit-failures`" - Successfully validated compare pairs are added to the repository and compare pairs that cannot be validated are ignored. "`none`" - Compare pairs are added to the repository without any validation. If this option is selected, the Oracle GoldenGate Veridata Web User Interface should be used to review and fix validation problems.
`operation`	Specifies how data is applied to the repository. The options are: "`create`" - All items listed in the configuration are new. If any item in the list exists in the repository, nothing is added. This can be used to prevent unintended modification to existing repository items. This is the *default* value. "`update`" - New items are added to the repository and existing items modified. Items existing in the repository and not listed in the configuration are deleted. "`delete`" - All named items in the configuration are removed from the repository. You can use a command line flag to override the value entered for this attribute.
`wildcard`	Specifies the pattern matching method that is used. The options are: "`ggs`" - Use the typical Oracle GoldenGate pattern using an asterisk (*). See the Oracle GoldenGate Veridata Web User Interface help for details on this type of matching. This is the default value. "`regex`" - Use regular expressions for matching.

Example

The following example adds compare pairs that can be validated and ignores those that cannot; uses regular expressions for wild carding; and uses the `"create"` default to adds all items as new items, adding nothing if any item already exists.

```
<configuration validation="omit-failures" wildcard="regex">
.
.
.
</configuration>
```

6.3.3.2 column

The `column` element defines a set of columns to be included or excluded from the compare pair. The `column` element has no nested elements or text data.

The following attributes describe the `column` element:

Table 6-3 column attributes

Attribute	Description
source-name	A regular expression that defines a set of source column names. This value is required.
target-name	A regular expression that defines a set of target column names. It can include references to groups captured by the source-name expression.
exclude	Indicates whether or not the matched columns should be excluded from the compare pair. The options are: "true" - The matched columns should be excluded. "false" - The matched columns should be included. This is the default.
type	Indicates the type of the column. The options are: "key" - The column is used as a key. "hash" - The column is compared using a hash value. This is the default value. "literal"- The column is a literal value.
format	Specifies a format to override the comparison format that would normally be used.
scale	Specifies a scale to override the default scale for the comparison.
precision	Specifies a precision to override the default precision used for the comparison.
timezone	Specifies a time zone to override the default time zone of the comparison.
optional	Indicates whether the column mapping is optional. For example, mapping will not fail if the base tables do not have the column patterns specified. Default is "false".

6.3.3.3 colfilter

The colfilter element defines a set of columns to be included or excluded. It is used to specify the names of the columns to use as filtering criteria.

The following element describes the colfilter element:

Table 6-4 colfilter Element

Attribute	Description
colfiltercol	Specifies a set of columns to be included or excluded.

The following attribute describes the colfilter element:

Table 6-5 colfilter Attribute

Attribute	Description
`type`	Specifies whether to include the columns or exclude them. The options are `include` or `exclude`; the default is `include`. This is a required attribute.

Example

This example excludes `COL3` and `COL5` for the table `TABLE_NAME` from the generated compare pair.

```
<filter type="include" table="TABLE_NAME">
    <colfilter type ="exclude">
        <colfiltercol name="COL3" />
        <colfiltercol name="COL5" />
    </colfilter>
</filter>
```

6.3.3.4 colfiltercol

The `colfiltercol` element defines a set of columns to be included or excluded. It is used to specify the names of the columns to use as filtering criteria.

The following attribute describes the `colfiltercol` element:

Table 6-6 colfiltercol Attribute

Attribute	Description
`name`	A regular expression that defines a set of source column names. This is a required attribute.

6.3.3.5 compare-pair

The `compare-pair` element specifies a set of compare pair items. As in the Oracle GoldenGate Veridata Web User Interface, the compare pairs default to system mapped keys and columns.

The following elements can be nested within the `compare-pair` element:

Table 6-7 compare-pair Elements

Element	Description
`enscribe-info`	One or more sets of information used when comparing NonStop Enscribe files.
`sql-partition`	One or more specifications of a subset of rows within the table.
`table-partition`	Specifies the database table partitions.
`enscribe-key`	One or more specifications of a subset of records within an Enscribe file.

Table 6-7 (Cont.) compare-pair Elements

Element	Description
`key-column`	A set of columns to be used as the user-defined key for the comparison.
`column`	One or more definitions of a set of columns to be included.
`excluded-column`	Defines a set of columns to excluded from the a compare pair when the compare pair uses system mapped columns.
`delta-config`	Defines the delta processing configuration for the compare pair. The maximum is to add it once per compare pair.

The following attributes describe the `compare-pair` element:

Table 6-8 compare-pair Attributes

Attribute	Element
`name`	An expression defining the name of the compare pair. This expression can include groups captured with `source-table` expressions and target table group `$0`.
`source-table`	A regular expression that defines the table or tables to be compared. See "Regular Expression Grouping" later in this section for more detail. The default is to match all tables.
`target-table`	A regular expression that defines the target tables for the comparison. This may contain references to groups captured by the source table expression. The default is $0 for the full source table name.
`source-schema`	The name of the default schema for the source tables referenced for the compare pair. The default is the value specified for the `group`. For SQL/MP, this is the subvolume of the SQL catalog. This is not used with Enscribe files.
`target-schema`	The name of the default schema for the target tables referenced for the compare pair. The default is the value specified for the `group`. For SQL/MP, this is the subvolume of the SQL catalog. This is not used with Enscribe files.
`source-catalog`	The default catalog for the source tables referenced in this compare pair. For SQL/MP, this is the volume of the SQL catalog. This is not used for the following databases: Oracle (Non-PDB), DB2, Enscribe, or Teradata. For Oracle PDB, this holds the PDB name.
`target-catalog`	The default catalog for the target tables referenced in this compare pair. For SQL/MP, this is the volume of the SQL catalog. This is not used for the following databases: Oracle (Non PDB), DB2, Enscribe, or Teradata. For Oracle PDB, this holds the PDB name.
`exclude`	Indicates whether or not the compare pair should be included in the `group` element. This can be used to remove a compare pair generated by an earlier compare pair element. The options are: `"true"` - Exclude the compare pair. `"false"` - Include the compare pair. This is the default.
`source-file-pattern`	The default file pattern for the source if the data source is Enscribe or SQL/MP.

Table 6-8 (Cont.) compare-pair Attributes

Attribute	Element
target-file-pattern	The default file pattern for the target if the data target is Enscribe or SQL/MP.
source-pkey	The name of the unique index to use as the source portion of the user-specified primary key. The default is no user-specified index name.
target-pkey	The name of the unique index to use as the target portion of the user-specified primary key. The default is the value of the source-pkey.
delta-processing	Indicates whether or not delta processing is enabled for this compare pair. The options are: "true" - delta processing is enabled. "false" - delta processing is not enabled. This is the default.
profile-name	The name of the profile to use when running the compare-pair comparison.
system-key	If the compare pair has no column elements and no specified source-pkey, Oracle GoldenGate Veridata will select the most appropriate primary key or unique index to use. The options are: "true" - Oracle GoldenGate Veridata selects the key if it is not defined. This is the default. "false" - Oracle GoldenGate Veridata does not select the key.
system-columns	Indicates that the compare pair contains column elements with the type attribute set to key, so the generated compare pair will have user-defined columns for the key. The options are: "true" - Compare pair has key column elements. This is the default. "false" - Compare pair does not have key column elements.
wildcard	Specifies the pattern matching method that is used. The options are: "ggs" - Use the typical Oracle GoldenGate pattern that matches an asterisk (*) to any number of characters. For Oracle GoldenGate HP Nonstop, the following is supported: source-table=""target-table="", and the following is not supported: source-table="TBL" target-table="". "regex" - Use regular expressions for matching. "default" - Use the setting for the configuration. This is the default.
is-auto	Specifies whether or not the Automatic Partitioned compare pairs need to be created in Oracle GoldenGate Veridata. "true" - Oracle Goldengate Veridata creates the auto partitioned compare pairs. The number of compare-pairs to be created is based on the value specified in no-of-auto-partitions. "false" - Oracle GoldenGate Veridata does not create auto-partitioned compare pairs. This is the default.
no-of-auto-partitions	Oracle GoldenGate Veridata creates the Auto Partitioned compare pairs. No of compare-pairs to be created is based on the value specified in no-of-auto-partitions.
use-source-keys	Defines key column as Source Key Columns OR Target key columns as key columns, when there are no keys selected for column mapping.
use-target-keys	Defines column values from target key to source columns, when there are no keys selected for column mapping.

Table 6-8 (Cont.) compare-pair Attributes

Attribute	Element
use-all-columns	Defines key column values from All Columns, when there are no keys selected for column mapping.

Regular Expression Grouping

Regular expression grouping can be used to capture the parts of the source table names to be used for matching the target table name. You can do this by changing the `wildcard` attribute should be changed to `regex`. Groups to be matched are referenced as `$1`, `$2`, `$3` and so on. Group `$0` matches the entire source table name.

Examples of matching groups include:

- `P(.*)` - Matches table names that begin with `P`. It captures the variable portion in `$1`. This matches table `PROSPECTS`.
- `[^PV].*` - Matches table names that do *not* begin with `P` or `V`. This does not match the table `PROSPECTS`, but does match the table `REGIONS`.
- `([P-R])(.*)` - Matches table names starting with `P`, `Q`, or `R` and captures the initial letter in group `$1` and the rest of the name in group `$2`. Groups are defined by parenthesis pairs. Group numbers are defined by the count of left parenthesis. Group `$1` starts at the first left parenthesis and group `$2` starts at the second parenthesis.

Captured groups (`$n`) are then used in expressions for selecting the target tables.

Example

The following example describes the `key-only` compare-pair. It's source tables are defined in the `"test"` schema and target tables in the `"other"` schema. It creates a compare pair in which the source table name begins with `S` and target table name begins with `T`. For example, `S_TABLE` and `T_TABLE`, where `S_TABLE` is a table in schema "test" and `T_TABLE` is table in schema "other". It also excludes all non-key columns in the generated compare pairs.

```
<configuration>
    <connection name="source" host="somehost"
               .... use-ssl="true">
        <description>
             <![CDATA[
                Group SQL Scripting Source Connection
             ]]>
          </description>
     </connection>
...
...
</configuration>
```

6.3.3.6 connection

The `connection` element defines a connection to a source or target comparison database through an Oracle GoldenGate Veridata agent.

The following elements can be nested within the `connection` element:

Table 6-9 connection Elements

Element	Description
`description`	Provides a description of the connection.
`conn-properties`	Defines the connection properties for a connection.

The following attributes describe the `connection` element:

Table 6-10 connection Attributes

Attribute	Description
`name`	A name that identifies the connection. This is a required attribute.
`host`	The name of the system on which the Oracle GoldenGate Veridata agent is running.
`port`	The port number of the system on which the agent is running.
`user`	The user name the agent uses to connect to the database.
`password`	The password the agent uses to connect to the database.
`repairUser`	The database user with privileges to perform repair operations. See Database Privileges for the Agent Component.
`repairPassword`	The password for the repairUser.
`agent-timeout`	The amount of time Oracle GoldenGate Veridata will wait before timing out when sending requests to the agent.
`truncate-spaces`	Either "`true`" or "`false`" to indicate whether or not spaces will be removed from the end of character columns. The default is "`true`" to truncate spaces.
`fetch-size`	(Oracle only) The number of rows fetched in each batch.
`use-ssl`	Defines using SSL communication between the Veridata Agent and the Server. The default is "`true`".
`use-source-keys`	Defines key column as Source Key Columns OR Target key columns as key columns, when there are no keys selected for column mapping.
`use-all-columns`	Defines key column values from All Columns, when there are no keys selected for column mapping.

Example

The following example identifies the connection named `source`.

```
<configuration>
   <connection name="source" host="somehost"
    port="7850" user="somename" password="somepw"repairUser="veridata1"
```

```
repairPassword="veridata1" agent-timeout="4000" truncate-spaces="false" fetch-size="3"
use-ssl="true">
      <description>
             <![CDATA[
                Group SQL Scripting Source Connection
            ]]>
...
...
      </description>
   </connection>
.
.
</configuration>
```

6.3.3.7 conn-properties

The `conn-properties` element provides additional connection to a source or target comparison database elements.

The following attributes can be nested within the `conn-properties` element:

Table 6-11 conn-properties

Element	Description
`datatype-name`	Specifies the data type for which properties have changed.
`format`	Specifies the Veridata comparison format to be used for comparison.
`precision`	Specifies the precision to be applied to the comparison.
`scale`	Specifies the scale to be applied to the comparison.
`timezone`	Timezone name is same as in the Veridata GUI.

6.3.3.8 delta-config

The `delta-config` element defines the delta processing configuration for the specified compare pair. It can be used once per compare pair. This element can appear once or not at all depending on the type of configuration you want. When the source or target configuration specified, the corresponding column-name attribute and query element are mandatory.

The following elements describe the `delta-config`:

Table 6-12 delta-config Elements

Attribute	Description
`source-config`	Provides source side configuration for delta processing.
`target-config`	Provides target side configuration for delta processing.
`query`	Specifies the query for delta processing.

Example

This example creates a compare pair with delta processing enabled. Delta processing is enabled on `COL1` of `SYSMAPPING1` table for both source and target side. The SQL query is defined within the "query" tag.

```
<configuration validation="required">
   .
   .
   <group name="testGroup" source-conn="sourceConn" target-conn="targetConn" source-
schema="sourceSchema" target-schema="targetSchema">
        <compare-pair source-table="SYSMAPPING1" target-table="SYSMAPPING1"
name="sameTables" delta-processing="true" >
                  <delta-config>
                        <source-config column-name="COL1">
                              <query><![CDATA[ SELECT MAX(COL1) from SYSMAPPING1 ]]></
query>
                        </source-config>
                        <target-config column-name="COL1">
                              <query><![CDATA[ SELECT MAX(COL1) from SYSMAPPING1 ]]></
query>
                        </target-config>
                  </delta-config>
            </compare-pair>
    </group>
      .
      .
</configuration>
```

6.3.3.9 description

The `description` element is free-form text that can be used to attach a description to the containing element. It has no associated attributes.

Example

The following example provides a description for the connection named `source`.

```
<configuration>
   <connection name="source" host="somehost"
    port="7850" user="somename" password="somepw"
      <description>
         <![CDATA[
           This connection is used when the Veridata agent connects
           to the source.
         ]]>
      </description>
   </connection>
.
.
.
</configuration>
```

6.3.3.10 enscribe-info

The `enscribe-info` element provides additional information used to compare NonStop Enscribe records at the field level.

The following elements can be nested within the `enscribe-info` element:

Table 6-13 enscribe-info Elements

Element	Description
`expandddl`	Describes the rules that are used when applying the DDL.

The following attributes describe the `enscribe-info` element:

Table 6-14 enscribe-info Attributes

Attribute	Description
`side`	Indicates whether the information applies to the source or the target table. The options are: `"source"` to specify the source table. This is the *default*. `"target"` to specify the target table.
`dictionary`	The volume and subvolume containing the data dictionary.
`record`	The name of the record in the data dictionary.

6.3.3.11 enscribe-key

The `enscribe-key` element defines the key that is to be used for Enscribe files. The `enscribe-key` element defines a delta processing that can used in a where clause on the initial comparison query.

The following attributes describe the `enscribe-key`:

Table 6-15 enscribe-key Attributes

Attribute	Description
`name`	A name that identifies the key. This is a required attribute.
`start-key`	The key that is to be used to begin reading the Enscribe file. This is a required entry.
`end-key`	The key of the last Enscribe record that should be read. This is a required entry.
`format`	Specifies the format of the Enscribe key. The options are: `"ascii"` - The format of the key is ASCII. This is the default. `"hexadecimal"` - The format of the key is hexadecimal.
`side`	Indicates whether the partition should be applied at the source database, the target database, or both databases.
`default`	Indicates whether this is the default partition. This is equivalent to the "use at run time" indicator on the UI. The default is `both`.

Examples

```
<enscribe-key name = "Part1" end-key ="1000" format ="hexadecimal" default ="false"
side="source"/>
<enscribe-key name = "Part1" start-key ="001" format ="hexadecimal" default ="false"
side="target"/>
<enscribe-key name = "Both" start-key ="001" end-key ="1000" default ="true"/>
```

6.3.3.12 excluded-column

The `excluded-column` element defines a set of columns to be excluded from a compare pair when the compare pair uses system mapped columns.

The following attribute describes the `excluded-column` element:

Table 6-16 excluded-column Attributes

Attribute	Description
`name`	A regular expression that defines a set of source column names. This is a required attribute.

6.3.3.13 expanddl

The `expanddl` element describes the rules used when applying the DDL.

The following attributes describe the `expanddl` element:

Table 6-17 expanddl Attributes

Attribute	Description
`expandGroupArrays`	Whether or not to expand group arrays. The options are: `"true"` to expand the array. This is the default. `"false"` not to expand the array.
`redefined-columns`	Whether or not to include redefined columns. The options are: `"include"` - Includes redefined columns `"omit"` - Leaves out redefined columns. This is the default.
`resolvedups`	Specifies how to resolve duplicates that result when the array is expanded. The options are: `"appendIndex"` - Adds a unique numeric index to the end of the duplicate. This is the default. `"appendAlphaIndex"` - Adds an alpha character index to the end of the duplicate. `"prependGroup"` - Prefixes the name of the array group to the duplicate.
`ddl-separator`	The character separator for defining array output into columns. An example is the dash used in FIELDX-3, which is the third occurrence of `FIELDX` in the array. The options are: `"none"` - There is no separator. This is the *default*. `"dash"` - Use a dash (-) as the separator. `"bracket"` - Use brackets [] as the separator. `"underscore"` - Use underscore (_) as the separator. `"double-underscore"` - Use double underscore (__) as the separator.
`zero-fill-length`	Prepends zeros to adjust the number of the occurrence. The value is the number of digits enclosed in quotation marks. "0" is the default.

Table 6-17 (Cont.) expandddl Attributes

Attribute	Description
fix-long-names	Whether to fix the names that result from resolving duplicates if they exceed the max-col-name-length. The options are: "true" - Fix the names that exceed the maximum. This is the default. "false" - Do not change the names that exceed the maximum.
max-col-name-length	The maximum length allowed for a column name. The entry is a number within quotation marks. The default is "120".

6.3.3.14 filter

The filter element defines a set of schemas and tables to either be included or excluded.

When using include filters, at least one filter must be matched before a table can be included in a compare pair. When a table matches a include filter, the include filter's colfilter is used to specify the columns for the generated compare pair.

When using exclude filters, a table is excluded if it matches any exclude filter. Include filters can include a colfilter element, which contains a list of columns to include or exclude.

Instead of schema and table filters, NonStop platforms use file pattern filters. The file pattern is any valid NonStop platform file name pattern.

The schema and table name can use wildcards.

The following attribute describes the filter element:

Table 6-18 filter Attributes

Attribute	Description
type	Specifies either to include or exclude schemas and tables. Valid values are include or exclude.
catalog	Specifies the default catalog name.
exclude-views	Excludes all the views while generating compare pairs. Valid values are true or false. Default value is false.
schema	Specifies the schema name.
table	Specifies the table name.
file-pattern	For NonStop platforms only, specifies the file patter filter.

Example

When the source and target schemas have CHAR_TYPES3, INT_TYPE1, and INT_TYPE2 tables, then the following filters only create compare pairs for tables CHAR_TYPES1 and CHAR_TYPES3. The CHAR_TYPES2 table is excluded because of exclude filter and INT_TYPE1 and INT_TYPE2 are excluded because they were not part of include filter.

```
<group
      ..
        <filter type="include" table="CHAR_TYPES*" />
        <filter type="exclude" table="CHAR_TYPES2" />
        <compare-pair source-table="*" target-table="*">
```

```
        </compare-pair>
      ..
</group>
```

6.3.3.15 group

The `group` element defines a set of compare pairs that all have the same source and target database connections. These compare pairs also have other properties in common.

The following elements can be nested within the `group` element.:

Table 6-19 Group Elements

Element	Description
`description`	Provides a description of the group.
`filter`	One or more filter specifications, which allows table name filtering at the group level.
`sql-partition`	One or more specifications of a subset of rows within the table.
`enscribe-key`	One or more specifications of a subset of records within an Enscribe file.
`compare-pair`	Defines one or more compare pairs. The `compare-pair` elements are added to the group in the order they are specified. If the same compare pair fits the criteria of another specification in the group, the first compare pair will be used.

The following attributes describe the `group` element:

Table 6-20 Group Attributes

Attribute	Description
`name`	A name that identifies the group. This value is required.
`source-conn`	The name of the connection to the source database. This can reference an existing connection in the Oracle GoldenGate Veridata repository or a connection previously defined in this configuration. This attribute is required if it references an existing connection in the repository.
`target-conn`	The name of the connection to the target database. This can reference an existing connection in the Oracle GoldenGate Veridata repository or a connection previously defined in this configuration. This attribute is required if it references an existing connection in the repository.
`source-schema`	The name of the default schema for the source tables referenced in the compare pairs that make up the group.
`target-schema`	The name of the default schema for the target tables referenced in the compare pairs that make up the group.
`source-catalog`	The default catalog for the source tables referenced in this group.
`target-catalog`	The default catalog for the target tables referenced in this group.

Table 6-20 (Cont.) Group Attributes

Attribute	Description
validation	Specifies the type of validation that will be used for the configurations. The options are: "required" - All compare pairs must be successfully validated before any pairs are added to the repository. "omit-failures" - Successfully validated compare pairs are added to the repository and compare pairs that cannot be validated are ignored. "none" - Compare pairs are added to the repository without any validation. If this option is selected the Oracle GoldenGate Veridata Web User Interface should be used to review and fix validation problems. "default" - Use the type of validation specified for a higher level, such as the configuration element. This is the default.
source-file-pattern	The default file pattern for the source if the data source is Enscribe or SQL/MP.
target-file-pattern	The default file pattern for the target if the data target is Enscribe or SQL/MP.

Example

```
<group name="weekly-tables" source-conn="source" target-conn"="target">
   <description>
   .
    .
    .
   </description>
   <sql-partition>
    .
    .
   </sql-partition>
   <compare-pair>
    .
    .
    .
</compare-pair>
</group>
```

6.3.3.16 job

The job element defines an Oracle GoldenGate Veridata comparison job.

The following elements can be nested within the job element:

Table 6-21 job Elements

Element	Description
description	Provides a description of the job.
group	The name of the group associated with the job. This can be a new group or a previously defined group.

The following attributes describe the `job` element:

Table 6-22 job Attributes

Attribute	Description
`name`	A name that identifies the job. This is a *required* attribute.
`source-conn`	The name of the connection to the source database. This can reference an existing connection in the Oracle GoldenGate Veridata repository or a connection previously defined in this configuration. This attribute is required if it references an existing connection in the repository. The job `source-conn` is used to override the source connection specified for the groups included in the job.
`target-conn`	The name of the connection to the target database. This can reference an existing connection in the Oracle GoldenGate Veridata repository or a connection previously defined in this configuration. This attribute is used to override the target connection for the groups included in the job.
`profile`	The default profile to use when running the job.

Example

```
<job name="all-groups" profile="server-sort">
    <group name="all-tables"/>
    <group name="selected-tables"/>
</job>
```

6.3.3.17 profile

The `profile` element defines the connection properties of a comparison job connection.

The following elements can be nested within the `profile` element:

Table 6-23 profile Elements

Element	Description
`description`	Provides a description of the profile.
`profile-general`	Defines the profile parameters that control the output options.
`sorting-method`	Defines the profile parameters that control the sorting method and memory management. The data is sorted to match keys (or a key specification) so that the correct source and target rows are compared.
`initial-compare`	Defines the profile parameters that control the parameters for the job that performs the initial compare step
`confirm-out-of-sync`	Specifies the profile parameters that control the parameters for the job that performs the confirmation step
`repair`	Specifies the profile parameters that control the parameters for the repair job.

The following attributes describe the `profile` element:

Table 6-24 profile Attributes

Attribute	Description
`name`	A name that identifies the profile. This is a required attribute.

Example

This example creates profile named "userDefinedProfile". The parameter names like "oos-format", "sort-method" are described in the table (link for table is in another pin)

```
<configuration validation="required">
   .
   .
<profile name="userDefinedProfile">
    <profile-general>
        <param name="oos-format" value="xml" />
        <param name="oos-xml-chunk-size" value="1000" />
    </profile-general>
    <sorting-method>
        <param name="sort-method" value="server" />
    </sorting-method>
</profile>
      .
      .
</configuration>
```

6.3.3.18 key-column

The `key-column` element defines a set of columns to be used as the user defined key for the comparison job.

The following attributes describe the `key-column` element:

Table 6-25 profile Attributes

Attribute	Description
`source-name`	A regular expression that defines a set of source column names. This value is required.
`target-name`	A regular expression that defines a set of target column names. It can include references to groups captured by the `source-name` expression.
`format`	Specifies a format to override the comparison format that would normally be used.
`scale`	Specifies a scale to override the default scale for the comparison.
`precision`	Specifies a precision to override the default precision used for the comparison.
`timezone`	Specifies a time zone to override the default time zone of the comparison.

6.3.3.19 profile-general

The `profile-general` element provides parameters to control the output options.

The data is sorted to match keys (or a key specification) so that the correct source and target rows are compared.

The following elements can be nested within the `profile-general` element:

Table 6-26 profile-general Element

Element	Description
`param`	Defines the parameter to change for the profile.

6.3.3.20 sorting-method

The `sorting-method` element provides parameters for sorting method and memory management. The data is sorted to match keys (or a key specification) so that the correct source and target rows are compared.

The following elements can be nested within the `sorting-method` element:

Table 6-27 sorting-method Element

Element	Description
`param`	Defines the parameter to change for the profile.

6.3.3.21 initial-compare

The `initial-compare` element provides parameters for the process that performs the initial compare step.

The following elements can be nested within the `initial-compare` element:

Table 6-28 initial-compare Element

Element	Description
`param`	Defines the parameter to change for the profile.

6.3.3.22 confirm-out-of-sync

The `confirm-out-of-sync` element provides parameters for the process that performs the confirmation step.

The following elements can be nested within the `confirm-out-of-sync` element:

Table 6-29 confirm-out-of-sync Element

Element	Description
`param`	Defines the parameter to change for the profile.

6.3.3.23 param

The `param` element defines the parameters that are used for configuring profile options.

The following attributes describe the `repair` element:

Table 6-30 param Attributes

Attribute	Description
`name`	The name of the parameter. This is a required attribute.
`value`	The value of the parameter

6.3.3.24 repair

The `repair` element provides parameters for the repair process.

The following elements can be nested within the `repair` element:

Table 6-31 repair Element

Element	Description
`param`	Defines the parameters that are used to configure the profile options.

6.3.3.25 sql-partition

The `sql-partition` element defines a boolean SQL expression that can be used in a `where` clause in the initial comparison query.

The following attributes describe the `sql-partition` element:

Table 6-32 sql-partition Attributes

Attribute	Description
`name`	A name that identifies the partition. This is a required attribute.
`side`	Indicates whether the partition should be applied at the source database, the target database, or both databases. The default is `"both"`.
`default`	Indicates whether this is the default partition. This is equivalent to the "use at run time" indicator on the UI. The default is `"false"`.

Table 6-32 (Cont.) sql-partition Attributes

Attribute	Description
type	Distinguishes between Manual and Automatic Row Partition. When type is set to sql, it defines Manual Partition and type is set to auto, defines Automatic Row Partition. This attribute is displayed during export of compare-pairs using Oracle GoldenGate Veridata Export tool.

Example

```
<sql-partition name="replicate" default="true" side="source" type="sql">
  <![CDATA[ replicated='false']]>
</sql-partition>
<sql-partition name="replicate" default="true" side="source" type="sql">
  <![CDATA[ replicated='true']]>
</sql-partition>
<sql-partition name="AutoPartition0" default="true" side="source" type="auto>
<![CDATA[2,0]]>
</sql-partition>
```

6.3.3.26 table partition

The table-partition element helps in specifiying the partition name details in the Oracle GoldenGate Export and Import tools.

For more information, see Mapping Database Table Partitions for Manual and Automatic Row Partitioning. To create compare-pairs using existing Database Table partitions, include the following lines in the XML when you execute the Import and Export tools:

```
<configuration operation="create" validation="required">
<group name="oracle_oracle_grp" source-conn="oracle_src" target-
conn="oracle_tgt">
 <compare-pair name="SALES_SRC_Q1_2006=SALES_GT_SALES_Q1_2006_AutoPartition0"
  source-table="SALES_SRC" target-table="SALES_TGT" source-schema="SYSTEM"
target-schema="SYSTEM"
  use-source-keys="true" use-target-keys="true" use-all-columns="true">

<table-partition name = "SALES_Q1_2006" default="true" side="target"/>
<table-partition name = "SALES_Q1_2006" default="true" side="target"/>
</compare-pair>
</group>
</configuration>
```

6.4 Server Parameters

Learn how to use parameters to adjust different aspects of the sort memory configuration when using server-side sorting.

This chapter includes the following sections:

- Overview of the Server Memory
- Estimating Memory Usage
- How to Set a Parameter

- Parameter Descriptions

6.4.1 Overview of the Server Memory

Oracle GoldenGate Veridata Server uses virtual memory in the following ways:

- **Server memory for basic operation**: This is the amount of virtual memory that the Veridata server and web components need to operate. It stores object pools, database access libraries, and other information. This is typically about 200 MB.
- **Sort memory**: This is the memory that is used when server-side sorting is used. The virtual memory for sorting is allocated for the entire comparison, not per thread. The rows are read from the agent and submitted to be sorted. The sorting occurs in a thread that is separate from the thread that reads from the agent, and the sort may use more threads to work in parallel. After all the rows from the agent are submitted to the sort process, the server process retrieves the sorted rows from the sort for comparison.
- **Row hash queue memory**: This is the memory that buffers data between the agent processes, the sort process, and the server process. A comparison that uses database sorting requires a single queue each for the source and target. Each queue has a capacity of 20 MG. The memory usage by the queues is affected by the relative speed of the comparison and by the data coming from the agent. The relative speed between the two agents also affects the memory usage. A larger differential in speed increases the amount of memory that is used, because the queue needs to buffer the data.
- **MOOS queue memory**: This is the memory that holds potentially out-of-sync records between the initial comparison and confirmation steps of a comparison. The size of the MOOS queue is limited to 50K of records. Memory usage is also dependent on the width of each record.
- **IPC buffer memory**: This is the memory that is used to exchange messages between the server and the agent.
- **Scratch runtime transient memory**: This is virtual memory space.

The amount of memory that can be used by the sort process cannot be greater than the minimum of:

- System physical memory
- Available memory in swap
- Java boot option `-Xmx` maximum memory setting

6.4.2 Estimating Memory Usage

The maximum amount of memory available to Oracle GoldenGate Veridata is specified by the Java boot option `-Xmx`. When the server-side sorting is used, a large portion of this memory is reserved for sorting during comparisons. This reserved amount is controlled by the `server.max_sort_memory` configuration parameter.

When a comparison is run, two buffers are allocated from the reserved sort memory. Each of these is equal to the size specified as Maximum Memory Usage (MB). To access this setting click the Edit option from the Profile Configuration screen, then Sorting Method from the Profile settings categories.

To Estimate Memory based on the Number of Concurrent Comparisons

The maximum amount of memory that can be used for any comparison is set by the parameter `server.max_comparison_sort_memory`. The `-Xmx` Java boot option should be set large enough to allow the desired number of concurrent comparisons.

The maximum number of concurrent comparisons is defined by the `server.max.concurrent_comparison_threads` configuration parameter. Therefore, the maximum amount of sort memory can be as large as: `server.max_comparison_sort_memory * server_max_comparison_threads`

For example, if you set `server.max_concurrent_comparison_threads` to allow 10 concurrent comparisons and leave `server.max_comparison_sort_memory` set to the default value of 100 MB, you will need 1 GB of available memory.

To Estimate the Amount of Memory Used per Row

For more information about the estimating the amount of memory used per row, see Disk and Memory Requirements for the Server Component.

6.4.3 How to Set a Parameter

To set a parameter, edit its entry in the `veridata.cfg` file. This file is stored in the `DOMAIN_HOME/config/veridata` directory within the Oracle GoldenGate Veridata Server installation directory.

Open an Oracle service request before changing these parameters.

6.4.4 Parameter Descriptions

This section describes the parameters that can be set in the `veridata.cfg` file. These parameters are grouped under the following categories:

- Server Parameters
- Parameters for Report File Encryption
- Parameters for Email Notifications and Alerts
- Server Parameters for Oracle Streams Tagging
- Server Parameters for Confirm Out-of-Sync Fetch Strategy

6.4.4.1 Server Parameters

This section defines the following configurable parameters for your Oracle GoldenGate Veridata Server:

- database.hash
- max_lob_key_len
- server.convert_empty_to_null
- server.mapped_sort_buffers
- server.max_concurrent_comparison_threads
- server.max_sort_memory
- server.memory_mapped_sort_directory
- server.meta_session_handle_timeout
- server.number_sort_threads
- server.veridata_data
- socket.timeout.mins

- server.useSsl
- truncate_spaces_len

6.4.4.1.1 database.hash

Syntax

Thi parameter defines whether or not the data hashing is done in the database. It can be used to improve the data-fetch performance.

```
database.hash = [true|false]
```

Default Value

```
false
```

> **Note:**
>
> This parameter is currently supported for the Oracle database only.

Example

```
database.hash = false
```

6.4.4.1.2 max_lob_key_len

This parameter sets the maximum length of LOB data types that can be used as key columns in Oracle GoldenGate Veridata. A positive integer enables the use of LOB data types as key columns. Note that this feature is currently only supported for Postgres' citext datatype. A value of zero disables this feature. The parameter value is measured in bytes, and must be within the range of 1 to 8000. If the size of the data in the table exceeds the defined value, then Oracle GoldenGate Veridata throws an exception and the job fails.

Syntax

```
max_lob_key_len = <length_of_lob_datatypes>
```

Default Value

```
0
```

Example

```
max_lob_key_len 1
```

6.4.4.1.3 server.convert_empty_to_null

If `server.convert_empty_to_null = false`, then Oracle GoldenGate for Veridata does not consider the implicit behaviour of compares empty (zero byte string) versus NULL (no space handling) to be insync

If `server.convert_empty_to_null = true`, then Oracle GoldenGate Veridata compares empty (zero byte string) versus NULL (no space handling) to be insync. If this is not comparison-format dependent, then one side should be Oracle and the other side being char, varchar, nchar, nvarchar, raw, clob, nclob, and blob datatypes of Oracle for which, this parameter is implicit.

Syntax

```
server.convert_empty_to_null = [true|false]
```

Default

```
true
```

Example

```
server.convert_empty_to_null = true
```

6.4.4.1.4 server.mapped_sort_buffers

Indicates whether sort buffers are allocated as a memory mapped file or allocated on the JVM heap.

Syntax

```
server.mapped_sort_buffers [true|false]
```

Example

```
server.mapped_sort_buffers true
```

Default Value

The default is `true`. If an error occurs during initialization, Oracle GoldenGate Veridata uses the JVM heap.

6.4.4.1.5 server.max_concurrent_comparison_threads

Syntax

This parameter sets the maximum number of concurrent comparisons that can be executed. You can lower this number to reduce the impact of the server on your system. When this limit is reached, no new comparisons, starts until an active comparison completes.

```
server.max_concurrent_comparison_threads {default | number}
```

Default Value

The maximim limit is 100. The default value is the maximum of four or the number of available CPUs.

Example

```
server.max_concurrent_comparison_threads 100
```

6.4.4.1.6 server.max_sort_memory

Syntax

This parameter sets the maximum amount of sort virtual memory that is available to all running comparisons that use server-side sorting. The value of this parameter varies depending on the value of `server.mapped_sort_buffers`.

```
server.max_sort_memory {default | number{M | m}}
```

`default` is illustrated in the follows:

- When `server.mapped_sort_buffers` is set to `true`, the default value is 2 GB. The maximum number supposed to enter is the maximum size of the memory mapped file.
- When `server.mapped_sort_buffers` is set to `false`, the default value is the JVM available heap size minus 200 MB, which is allocated for basic tasks. The maximum number supposed to enter is the value of `java -Xmx`.

Example

```
server.max_sort_memory 1000M
```

Default Value

The system calculates the default size based on the available virtual memory.

6.4.4.1.7 server.memory_mapped_sort_directory

Syntax

This parameter specifies the directory path where `.map` files are generated. By default, it uses temp directory of the operating system. If your operating system temp directory is less than 50GB, then you should consider specifying a larger directory.

```
server.memory_mapped_sort_directory directorypath
```

Example

```
server.memory_mapped_sort_directory u01/tmp2
```

6.4.4.1.8 server.meta_session_handle_timeout

This parameter defines the meta-session handle timeout in seconds.

Syntax

```
server.meta_session_handle_timeout seconds
```

Example

```
server.meta_session_handle_timeout 600
```

Default Value

900

6.4.4.1.9 server.number_sort_threads

This parameter specifies the number of threads used to sort input buffers from the Veridata Agent.

> **Note:**
>
> The value of `server.number_sort_threads` should not be greater than the number of available processes.

Syntax

```
server.number_sort_threads number
```

Example

```
server.number_sort_threads number
```

Default Value

The maximum of 4 or one quarter of the number of available CPUs.

6.4.4.1.10 server.veridata_data

The directory that contains Oracle GoldenGate Veridata reports.

Syntax

```
server.veridata_data path
```

where *path* is a relative or absolute path for the directory where Veridata reports will be stored.

> **Note:**
>
> If you specify a relative path for the data directory, you need not start the path with a forward (/) or backward (\) slash. The path will be relative to the Veridata domain home directory.

Default Value

```
veridata/reports
```

Example

```
server.veridata_data /u01/veridata/reports
```

This means change output directory from default `VERIDATA_HOME/veridata/reports` to `/u01/veridata/reports`.

6.4.4.1.11 socket.timeout.mins

This is used to set the timeout (in minutes) for inactive sockets.

Syntax

```
socket.timeout.mins <minutes>
```

Default Value

```
0
```

Example

```
socket.timeout.mins 0
```

6.4.4.1.12 server.useSsl

This parameter specifies whether SSL is enabled for communication between the Veridata Server and all Veridata Agents.

Syntax

```
server.useSsl [true|false]
```

Example

```
server.useSsl true
```

Default Value

The default value is false.

6.4.4.1.13 truncate_spaces_len

Use the `truncate_spaces_len` parameter to control how trailing spaces in a String or Binary column are truncated. Space (U+0020) and Ideographic Space (U+3000) are the truncate targets.

You can disable the truncate feature with the **Truncate Trailing Spaces When Comparing Values** parameter of the **Connection Settings** tab in the Oracle GoldenGate Veridata UI of the connection configuration.

This parameter also truncates the column padding character if the padding character is one of the target spaces.

If the value equals 0, then all spaces are trimmed. For example, data = "".

If the value equals 1, then at least one space gets retained. For example data = " ".

Syntax

```
truncate_spaces_len <number_of_trailing_spaces>
```

Default Value

```
1
```

Example

```
truncate_spaces_len 0
```

6.4.4.2 Parameters for Report File Encryption

This section defines the configurable parameters used for report file encryption:

- server.encryption
- server.encryption.bits

6.4.4.2.1 server.encryption

When this parameter is set to true, the comparison report artifacts will be encrypted. Otherwise, the report contents will be in clear text.

Syntax

```
server.encryption=[true|false]
```

Example

```
server.encryption=false
```

Default Value

The default value is false.

6.4.4.2.2 server.encryption.bits

This parameter specifies the strength of the encryption algorithm. Valid values are 128, 192, and 256. If set to a value other than 128, you must install Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files.

Syntax

```
server.encryption.bits=[128|192|256]
```

Example

```
server.encryption.bits=128
```

Default Value

The default value is 128.

6.4.4.3 Parameters for Email Notifications and Alerts

To send email notifications, Oracle GoldenGate Veridata can connect to an outgoing Simple Mail Transfer Protocol (SMTP) server. The SMTP server needs to be configured so it does not require password credentials.

If password is required, then see mail.from.

This topic describes the parameters required for email notifications and alerts:

- mail.enabled
- mail.alert.only
- mail.smtp.server
- mail.smtp.port
- mail.smtp.connection.ssl.tls
  The `mail.smtp.connection.ssl.tls` parameter is used to access mail server over SSL or TLS connection.
- mail.from
- mail.to
  The `mail.to` parameter is used to send emails to one or multiple users. Use a comma as a separator between email IDs.

6.4.4.3.1 mail.enabled

When this parameter is set to `false`, the email notifications are turned off.

Syntax

```
mail.enabled=[true|false]
```

Example

```
mail.enabled=false
```

Default Value

The default value is `false`.

6.4.4.3.2 mail.alert.only

When this parameter is set to `false`, Oracle GoldenGate Veridata sends a notification email when a job is completed. If it is set to `true`, then Veridata only sends an alert email when out of sync or failure are encountered.

Syntax

```
mail.alert.only=[true | false]
```

Example

```
mail.alert.only=true
```

Default Value

The default value is `true`.

6.4.4.3.3 mail.smtp.server

Syntax

```
mail.smtp.server=<server_address>
```

Example

```
mail.smtp.server=smtp.host.com
```

6.4.4.3.4 mail.smtp.port

Syntax

```
mail.smtp.port=<port_number>
```

Example

```
mail.smtp.port=25
```

6.4.4.3.5 mail.smtp.connection.ssl.tls

The `mail.smtp.connection.ssl.tls` parameter is used to access mail server over SSL or TLS connection.

Syntax

```
mail.smtp.connection.ssl.tls= [true|false]
```

Default Value

```
false
```

Example

```
mail.smtp.connection.ssl.tls=true
```

6.4.4.3.6 mail.from

This parameter is a username created on SMTP server that does not require a password authentication. If password is required, create a generic user with default password Ver! data1234 (case sensitive).

Syntax

```
mail.from=<email_address>
```

Example

```
mail.from=john@mail.com
```

6.4.4.3.7 mail.to

The `mail.to` parameter is used to send emails to one or multiple users. Use a comma as a separator between email IDs.

Syntax

```
mail.to=<email_address(es)>
```

Example

```
mail.to=john@mail.com,mary@mail.com,ross@mail.com
```

6.4.4.4 Server Parameters for Oracle Streams Tagging

Oracle GoldenGate Veridata tags the repair transaction using Oracle Stream API. Following properties are added as server parameters to customized the tags behavior:

- repair.oracle.tag.enable
- repair.oracle.tag

6.4.4.4.1 repair.oracle.tag.enable

This parameter enables tagging of Repair session in the redo logs.

Valid only for Oracle Database.

Syntax

```
repair.oracle.tag.enable = false
```

Value can be either `true` or `false`.

Default Value

```
true
```

6.4.4.4.2 repair.oracle.tag

This parameter is the tag value of Repair session in the redo logs.

Valid only for Oracle Database.

Syntax

```
repair.oracle.tag = 00
```

Value can be up to 2000 hexadecimal digits (0-9A-F) or the plus sign (+).

Default Value

```
00
```

6.4.4.5 Server Parameters for Confirm Out-of-Sync Fetch Strategy

This topic contains the following parameter:

- coos.join.strategy

6.4.4.5.1 coos.join.strategy

Internally, COOS with JOIN uses creation of temporary table. For this you need `CREATE TABLE`, `CREATE TABLESPACE`, `CREATE PRIVATE TEMPORARY TABLE`, and `INSERT/UPDATE/DELETE` privileges

Syntax

```
coos.join.strategy = nokey
```

Value can be `nokey`, `always`, or `never`.

Description

`nokey` - Use COOS Join strategy only when Use all columns as Key columns (there is no primary or unique constraint on the table) is used to select key columns for the compare pair. All columns as key columns in the compare pair exception XML, UDT and Lobs.

`always` - Use COOS Join always for COOS step.

`never` - Don't use COOS Join for COOS Step.

Default Value

```
nokey
```

6.5 Agent Parameters - General

This topic defines the following configurable parameters for your Oracle GoldenGate Veridata Agent:

- compare.xmldatatype.format
- coos.batch.fetch
- database.characterSet
- database.transaction.isolation
- pool.checkInterval
- pool.maxIdle
- pool.maxSize
- pool.maxIdleTime
- pool.maxStatements
- server.port
- rowscn
- zlib.buffer.flush.size

6.5.1 compare.xmldatatype.format

To use the `INDENT/NO INDENT` functionality of XMLSerialize, update the `compare.xmldatatype.format` to either `true` or `false`. By default, the comparison is done with `INDENT`. To select `NO INDENT` set `compare.xmldatatype.format=false`.

> **Note:**
>
> The `compare.xmldatatype.format` parameter is only for Oracle Agent.

Syntax

```
compare.xmldatatype.format=true
```

Default

```
true
```

6.5.2 coos.batch.fetch

Parameter to fetch coos data in batches.

Default Value

```
false
```

6.5.3 database.characterSet

The `database.characterSet` parameter is used for overriding the source database character for comparison.
The parameter value should be the name of the character set used to encode the character data (CHAR, VARCHAR2, CLOB and LONG).

This parameter should match the `SOURCECHARSET OVERRIDE` in the replicat parameter file at the target database.
This is supported only for Oracle databases.

6.5.4 database.transaction.isolation

The `database.transaction.isolation` property controls the transaction isolation level used during initial compare.
The default value for Sybase, DB2, SQL Server and Teradata is `READ_UNCOMMITTED`.

The only value supported for Oracle is `READ_COMMITTED`.
SQL Server versions 2005 above also support the value `SNAPSHOT` which requires that `ALLOW_SNAPSHOT_ISOLATION` is enabled in the database.

Confirm out of sync always uses the `READ_COMMITTED` transaction isolation level.

Default Value

```
database.transaction.isolation=READ_UNCOMMITTED
```

6.5.5 pool.checkInterval

Time interval to check the timeouts for idle connections.

Default Value

1/4th of `pool.maxIdleTime`

6.5.6 pool.maxIdle

Max number of idle database connections present in pool.

Default Value

```
20
```

6.5.7 pool.maxSize

Database connection pool maintained by Veridata. It defines the max number of database connections held by this pool. There will a separate pool maintained by Veridata for each user and for each PDB in-case of Container DB.

Default Value

```
20
```

6.5.8 pool.maxIdleTime

Timeout for idle connections in the connection pool. The value is in seconds.

Default Value

`300`

6.5.9 pool.maxStatements

The number of statements cached per connection. The default value is 20.

Default Value

`20`

6.5.10 server.port

The `server.port` property is the port where the Oracle GoldenGate Veridata Agent listens for connection requests.

Syntax

`server.port=`*`Port Number`*

For example:

```
server.port=7862
```

6.5.11 rowscn

Use this property if you want to skip initial delta comparison. Only the rows greater than the given SCN value will be compared.
This property is enabled only for Oracle Agent.

> **Note:**
>
> This property works either with or without delta. However if you enable delta, then ensure that `ORA_ROWSCN` is used as delta column which is the default delta column for Oracle tables.

Syntax

`rowscn=<`*`scn number`*`>`

Default Value

None

6.5.12 zlib.buffer.flush.size

The `zlib.buffer.flush.size` property is used to decide the `extra flush()` call.

Decrease the value to a lower value (for example, `700000`) if the agent appears to be in hanged state and the CPU usage for agent process is about 100%.

The value should be between `10000` and `2000000`.

Default Value

`1000000`

6.6 Agent Parameters - Connections

- DB2iseries
- DB2luw
- DB2zos
- Hive
- MSSQL
- MySQL
- Oracle
- PostgreSQL
- Sybase
- Teradata

6.6.1 DB2iseries

DB2 i sample database connection URL

LocationName is usually the i server name.

```
database.url=jdbc:veridata:db2://DB2HOST:446;LocationName=DB2HOST
server.jdbcDriver=vddb2-5.1.4.jar
```

Settings for using the JTOpen driver for DB2 i

```
database.url=jdbc:as400:DB2HOST
server.jdbcDriver=jt400Native.jar
```

Settings for using the IBM Native driver for DB2 i

```
database.url=jdbc:db2i:DB2HOST:portNumber=446;preFetch=true
server.jdbcDriver=db2_classes.jar
```

6.6.2 DB2luw

DB2 LUW sample database connection URL

```
database.url=jdbc:veridata:db2://localhost:50000;DatabaseName=sample
server.jdbcDriver=vddb2-5.1.4.jar
```

6.6.3 DB2zos

DB2 z/OS sample database connection URL

```
database.url=jdbc:veridata:db2://localhost:447;LocationName=DB2
server.jdbcDriver=vddb2-5.1.4.jar
```

6.6.4 Hive

Hive sample database connection URL for Apache Hive2 JDBC Driver

```
database.url=jdbc:hive2://localhost:10000
server.jdbcDriver=hive-jdbc-3.1.2-standalone.jar
```

6.6.5 MSSQL

SQL Server sample database connection URL for SSL/TLS

```
database.url=jdbc:sqlserver://
localhost:1443;databaseName=<dbName>;encrypt=false;trustServerCertificate=false;
server.jdbcDriver=mssql-jdbc-11.2.0.jre17.jar
```

SQL Server database connection URL

```
database.url=jdbc:veridata:sqlserver://localhost:1433
server.jdbcDriver=vdsqlserver-6.0.0.jar
```

6.6.6 MySQL

MySQL sample database connection URL

```
database.url=jdbc:mysql://localhost:3306/<db_name>?serverTimezone=UTC&
zeroDateTimeBehavior=CONVERT_TO_NULL&sessionVariables=sql_mode='PAD_CHAR_TO_FULL_LENGTH'
server.jdbcDriver=mysql-connector-j-8.3.0.jar
```

MySQL sample JDBC URL for MySQL Server Authentication via server certificate

(For MySQL version 8.0.12 and earlier):

```
database.url=jdbc:mysql://localhost:3306?useSSL=true&verifyServerCertificate=true
server.jdbcDriver=mysql-connector-j-8.3.0.jar
```

For MySQL version 8.0.13 and later:

```
database.url=jdbc:mysql://localhost:3306?sslMode=<VERIFY_CA or VERIFY_IDENTITY>
server.jdbcDriver=mysql-connector-j-8.3.0.jar
```

6.6.7 Oracle

Oracle sample database connection URL

```
database.url=jdbc:oracle:thin:@localhost:1521:orcl
database.url=jdbc:oracle:thin:@localhost:1521/PDB_service_name
server.jdbcDriver=ojdbc11-23.2.0.0.jar
```

Oracle OCI bequeath database connection URL

The OCI libraries must be available and the JDBC driver must match the OCI libraries.

```
database.url=jdbc:oracle:oci:@
server.jdbcDriver=ojdbc11-23.2.0.0.jar
```

Oracle sample database connection URL for SSL/TLS

```
database.url=jdbc:oracle:thin:@tcps://<host>:<port>/<service name>?
wallet_location=<wallet directory path>
database.url=jdbc:oracle:thin:@tcps://localhost:2484/service_name?wallet_location=/path/
WALLET
server.jdbcDriver=ojdbc11-23.2.0.0.jar
```

6.6.8 PostgreSQL

Postgresql sample database connection URL for native driver

```
database.url=jdbc:postgresql://localhost:5432/target
server.jdbcDriver=postgresql-42.7.4.jar
```

Postgresql sample database connection URL for SSL/TLS

```
database.url=jdbc:postgresql://<host>:5432/postgres?sslmode=verify-
ca&sslrootcert=<crt_path>/<crt_filename>
server.jdbcDriver=postgresql-42.7.4.jar
```

6.6.9 Sybase

Sybase database connection URL

`JDBCBehavior=0` is required to repair Sybase `UNITEXT` columns.

```
database.url=jdbc:veridata:sybase://
localhost:5000;ApplicationName=VeriAgent;MaxPooledStatements=20;
JDBCBehavior=0
server.jdbcDriver=vdsybase-5.1.4.jar
```

6.6.10 Teradata

Teradata database connection URL

```
database.url=jdbc:teradata://localhost/DBS_PORT=1025,CHARSET=UTF8
server.jdbcDriver=terajdbc4.jar
```

7
Secure

7.1 Configuring Workflow for Two-Way SSL in Oracle GoldenGate Veridata 23c

Prerequisites

- Ensure PATH environment variable contains the path to JDK 17 bin directory, where keytool is located.
- Keystore is also known as Identity Store in older releases.
- Veridata 23.1.0.0.0 NSK C-Agent currently supports only one-way SSL. See Configuring One-Way SSL for the NSK C-Agent on the Veridata Server.

This topic comprises the following:

7.1.1 Enabling SSL in the Agent Properties File

To enable SSL in the Agent properties file:

1. Open the *<AGENT_DEPLOY_LOCATION>/<AGENT PROPERTIES>* file. The default *<AGENT PROPERTIES>* would be `agent.properties`.

2. Find the entry `server.useSsl`, set it to `true` and save the changes.

7.1.2 Generating Agent Keystore and Certificate

To generate agent keystore and certificate file:

1. Go to `<AGENT_DEPLOY_LOCATION>/config/certs` directory.
2. Build Agent Keystore with the following command, you will also need the same unlock for later step.

```
keytool -genkeypair -keyalg RSA -keystore vdtAgentKeystore.p12 -storepass
<unlock-password>
```

3. Export Agent Keystore to a certificate with following command:

```
keytool -exportcert -keystore vdtAgentKeystore.p12 -storepass <unlock-
password> -file vdtAgent.crt
```

7.1.3 Generating Server Keystore and Certificate

Use the following keytool commands to generate the server keystore and certificate:

1. Go to *<Server_installation_location>*`/config` directory.
2. Build Server Keystore with the following command:

```
keytool -genkeypair -keyalg RSA -keystore vdtServerKeystore.p12 -storepass
<unlock-password>
```

3. Export Server Keystore to a certificate with the following command:

```
keytool -exportcert -keystore vdtServerKeystore.p12 -storepass <unlock-
password> -file vdtServer.crt
```

7.1.4 Importing Agent Certificate to Server Truststore

To import agent certificate to Server truststore:

1. Go to *<Server_installation_location>*`/config` directory, and copy *<AGENT_DEPLOY_LOCATION>*`/config/certs/vdtAgent.crt` to this directory.
2. Run the following command to create a server truststore and import agent certificate into this truststore.

> **Note:**
>
> When importing multiple agent certificates into a server truststore, assign a unique `-alias` value to each agent certificate.

```
keytool -importcert -file vdtAgent.crt -alias vdtAgent.crt.<unique-id> -
keystore vdtServerTruststore.p12 -storepass <unlock-password>
```

3. Delete `vdtAgent.crt` in *`<Server_installation_location>`*`/config` directory.

7.1.5 Saving Server Keystore/Truststore Passwords to Server Wallet

Run the following script `configure_server_ssl.sh` under *`<Server_installation_location>`*`/config` directory:

```
./configure_server_ssl.sh
OGGV-80056: Copyright (c) 2013, 2024, Oracle and/or its affiliates. All rights reserved.
OGGV-80057: Veridata Server SSL Configuration Utility
OGGV-80058: Notes:
OGGV-80059: OGGV-80059: This utility allows Veridata to access keystore and truststore.
When entering the passwords below, unlock password should match the one used in keytool -
storepass option
[OGGV-80060: Enter Server Keystore unlock password:]
[OGGV-80062: Enter Server Truststore unlock password:]
```

7.1.6 Importing Server Certificate to Agent Truststore

To import Server certificate to agent truststore:

1. Go to *`<AGENT_DEPLOY_LOCATION>`*`/config/certs` directory, and copy *`<Server_installation_location>`*`/config/vdtServer.crt` to this directory.
2. Run the following command to create an agent truststore and import server certificate into this truststore.

> **Note:**
>
> When importing multiple Server certificates into an agent truststore, assign a unique `-alias` value to each Server certificate. This is less common, because there is typically only one Oracle GoldenGate Veridata server.

```
keytool -importcert -file vdtServer.crt -alias vdtServer.crt.<unique-id> -
keystore vdtAgentTruststore.p12 -storepass <unlock-password>
```

3. Delete `vdtServer.crt` in *`<AGENT_DEPLOY_LOCATION>`*`/config/certs` directory.

7.1.7 Saving Agent Keystore/Truststore Passwords to Agent Wallet

1. Run the script `configure_agent_ssl.sh` under *<AGENT_DEPLOY_LOCATION>* directory. The parameter `AgentID` is the name of the agent properties file, without the `.properties` extension.:

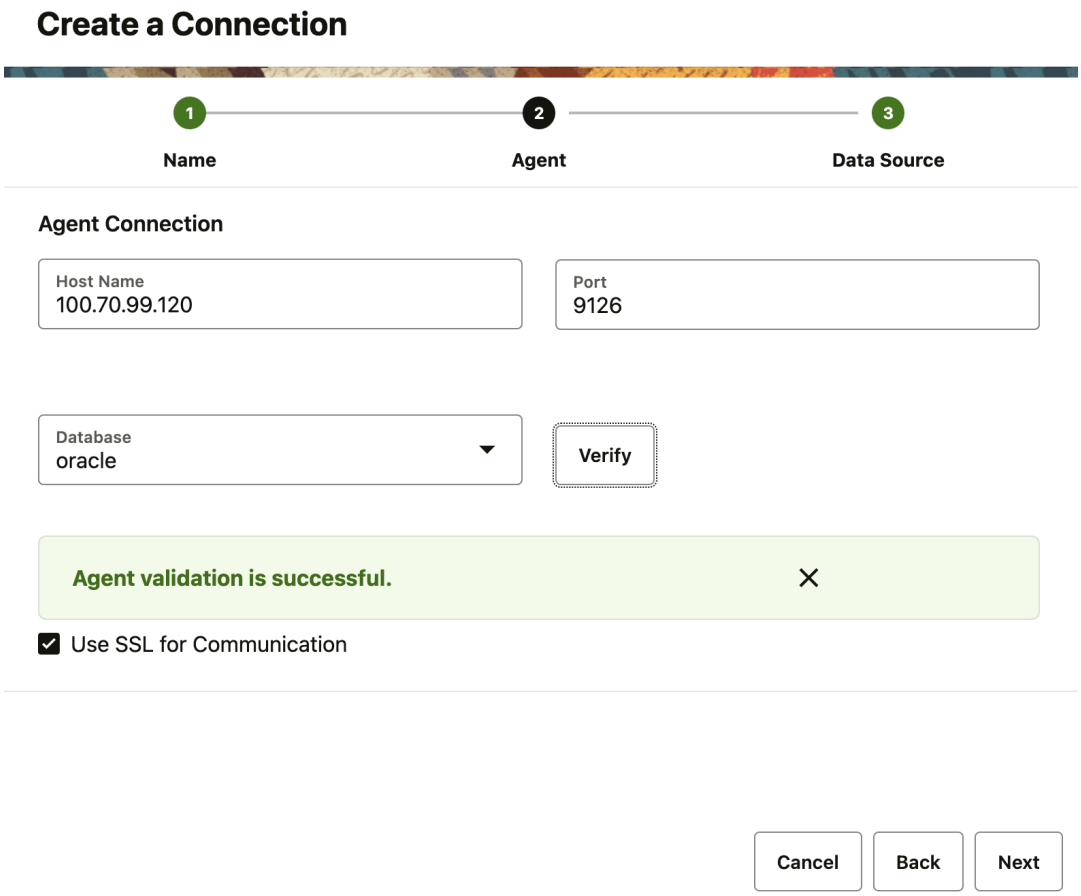
```
./configure_agent_ssl.sh AgentID
2024-08-30 11:21:25.782 TRACE OGGV-80018 Wallet messages are installed correctly.
OGGV-80028: Copyright (c) 2013, 2024, Oracle and/or its affiliates. All rights
reserved.
OGGV-80029: Veridata Agent SSL Configuration Utility
OGGV-80030: Notes:
OGGV-80031: OGGV-80031: This utility allows Veridata to access keystore and
truststore. When entering the passwords  below, unlock password should match the one
used in keytool -storepass  option.
[OGGV-80022: Enter Agent Keystore Store unlock password:]
[OGGV-80024: Enter Agent Trust Store unlock password:]
OGGV-80037: SSL Configuration of Veridata Agent is successful.
```

2. Delete `vdtServer.crt` in *<AGENT_DEPLOY_LOCATION>*`/config/certs` directory.

7.1.8 Creating an Agent Connection in UI

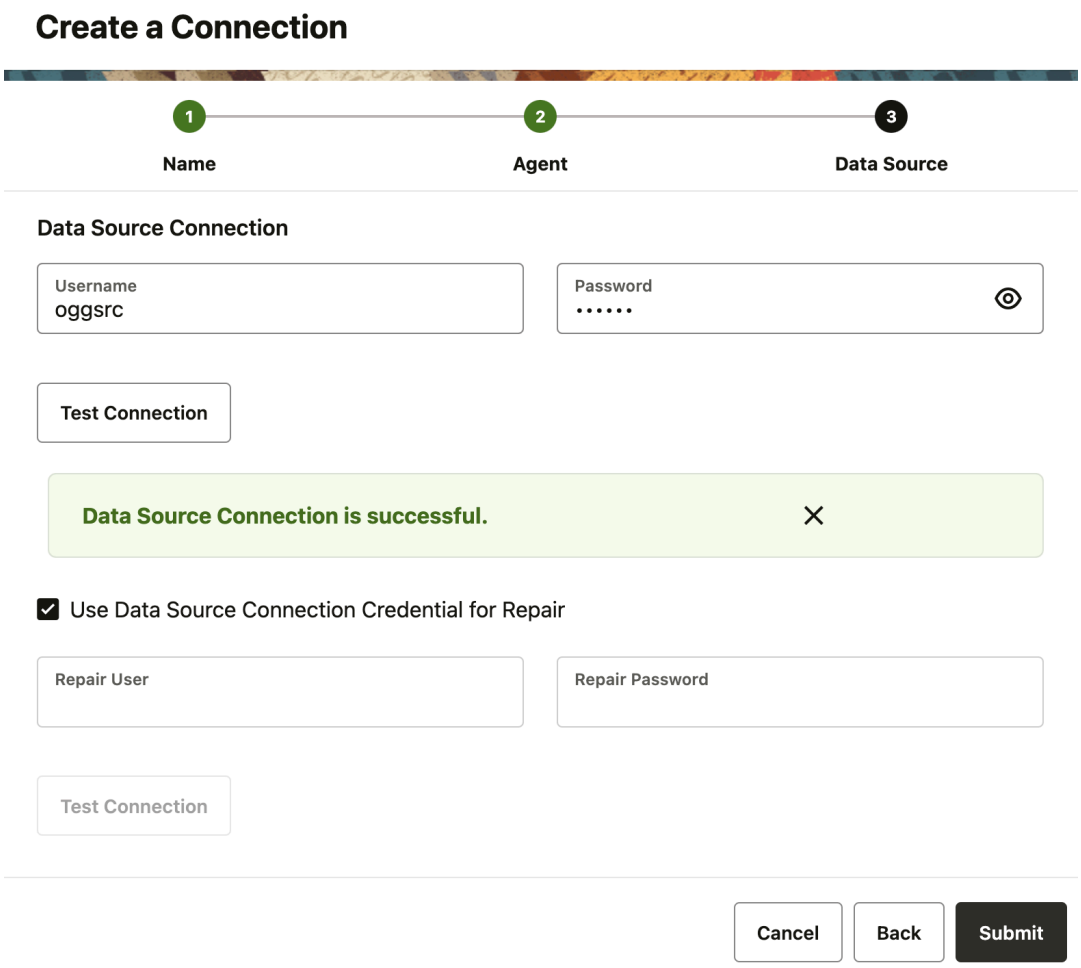
1. Log in to the Veridata website, navigate to the **Connections** page, and click **Create** on the right side of the page.
2. Enter the required connection name and agent host machine address/port. Check the Use SSL for communication checkbox to enable SSL for the connection, and click **Verify** to verify the connection:

Figure 7-1 Create a Connection



3. Enter the database username and password, then click the **Test Connection** to validate the credentials. If different credentials are needed for repair, uncheck the **Use Data Source Connection Credential for Repair** checkbox and enter the repair credentials.

Figure 7-2 Data Source Connection



4. Click **Submit** to save the connection.

7.2 Configuring One-Way SSL for the NSK C-Agent on the Veridata Server

If the Oracle GoldenGate Veridata server keystore has not been created, then use the keytool command as follows to generate the server keystore.

1. Go to *<Server_installation_location>*/config directory.
2. Build Server Keystore with the following command:

```
keytool -genkeypair -keyalg RSA -keystore vdtServerKeystore.p12 -storepass
<unlock-password>
```

Import NSK C-Agent Certificate to Server Truststore

1. Go to *<Server_installation_location>*/config directory, and copy the NSK C-agent certificate to this directory.

2. Run the following command to create a server truststore and import the NSK C-agent certificate into this truststore:

```
keytool -importcert -file <nsk-agent-certificate-file> -alias
vdtAgent.crt.<unique-id>  -keystore vdtServerTruststore.p12 -storepass
<unlock-password>
```

> **Note:**
>
> When importing multiple NSK C-agent certificates into a server truststore, assign a unique `-alias` value to each NSK C-agent certificate.

3. Delete the NSK C-agent certificate in *<Server_installation_location>*`/config` directory.

Save Server Truststore passwords to Server Wallet

1. Run the script named `configure_server_ssl.sh` under *<Server_installation_location>*`/config` directory.
2. For One-Way SSL, a Server Keystore is not required. If no Server Keystore has been configured, then press Enter when prompted. However, if a Server Keystore is available, enter the password when prompted.

```
./configure_server_ssl.sh
OGGV-80056: Copyright (c) 2013, 2024, Oracle and/or its affiliates. All rights
reserved.
OGGV-80057: Veridata Server SSL Configuration Utility
OGGV-80058: Notes:
OGGV-80059: This utility allows Veridata to access keystore and truststore. When
entering the passwords below, unlock password should match the one used in keytool -
storepass option.
[OGGV-80060: Enter Server Keystore unlock password:]
[OGGV-80062: Enter Server Truststore unlock password:]
```

7.3 Configuring Oracle GoldenGate Veridata Agent Using Kerberos to Connect to Oracle Database

To configure Oracle GoldenGate Veridata Agent using Kerberos to connect to Oracle database:

1. Complete the steps detailed in #unique_237.
2. Initiate initial ticket granting ticket for the principal using `okinit`. To request an initial ticket, run `okinit username`. The `username` is the user created or configured to use kerberos.
3. Login database instance with an **Oracle Net Service** service name. Run `sqlplus / @service_name` to login to the db instance, and then run `show user`. The displayed user should be the user granted the initial ticket before.
4. Copy Kerberos configuration file and ticket cache file into Veridata agent deploy directory. Absence of either file in agent deploy directory disables the kerberos use of the Oracle GoldenGate Veridata agent.
5. Edit `agent.properties`. For example: `database.url=jdbc:oracle:thin:@host1.us.oracle.com:1522:vdtkbr`. The `database.url` is the same as the url that is in a non-Kerberos configuration.

6. In the `agent.properties` file, add, uncomment, and edit the entries, `kerberos.configuration.file.name` and `oracle.kerberos.ticket.cache.file.name`. A missing entry or an incorrect entry disables the Kerberos use of Oracle GoldenGate Veridata agent. If Kerberos use is not desired, then comment out or delete either of the entries.

   For example:

   ```
   #Kerberos configuration file name.
   Comment the entry to disable veridata agent to use kerberos.
   #To make veridata agent to use kerberos, the file must be in the agent install
   directory.
   kerberos.configuration.file.name=krb.conf
   #Kerberos ticket cache file name for Oracle.
   #To make veridata agent to use kerberos, the file must be in the agent install
   directory.
   oracle.kerberos.ticket.cache.file.name=krb.cc
   ```

7. Start the Oracle GoldenGate Veridata Agent: `./agent.sh`.
8. Verify connection in UI. Note that you do not have to enter the username and password in the Database details.

7.4 Configuring Oracle GoldenGate Veridata Agent Using Kerberos to Connect to Hive

To configure Oracle GoldenGate Veridata Agent using Kerberos to connect to Hive database:

1. Complete the steps detailed in #unique_237.
2. Obtain the Kerberos configuration file from the kerberos server, for example, `krb5.conf`.
3. Copy the Kerberos configuration file in the OS default location. For example, in Linux, it is `/etc/`.
4. For Hive specific configurations, review: `VERIDATA_AGENT_HOME/veridata/agent/sample_properties/agent.properties.hive`.
5. Obtain the the keytab file from Kerberos enabled Hive environment.
6. Copy the keytab file into the Oracle GoldenGate Veridata Agent deploy directory.
7. In the `agent.properties` file, add/uncomment, and edit the entries `hive.kerberos.keytab.file.name` and enter the keytab file name.
8. Edit `agent.properties` and add the `database.url`. For Kerberos authentication principal argument is required. For example, in a Cloudera Hive, following is the jdbc url syntax: .

   ```
   database.url=jdbc:hive2://<Hive server host>:10000/default;principal=hive/
   HiveServerHost@YOUR-REALM.COM
   ```

9. Edit `agent.properties.hive` and add the `server.jdbcDriver` appropriately. For example, for Cloudera Hive:

   ```
   server.jdbcDriver=commons-collections-3.2.1.jar hadoop-common-2.4.1.jar
   hive-service-0.14.0.jar hadoop-mapreduce-client-core-2.4.1.jar hive-shims-
   common-0.14.0.jar
   commons-logging-1.1.3.jar hive-exec-0.14.0.jar log4j-1.2.17.jar hive-jdbc-0.14.0-
   standalone.jar
   slf4j-api-1.7.5.jar hadoop-auth-2.4.1.jar hive-metastore-0.14.0.jar slf4j-
   log4j12-1.7.5.jar
   commons-configuration-1.6.jar commons-dbcp2-2.5.0.jar commons-pool2-2.6.0.jar
   ```

Obtain the appropriate versions of these jars from Hive environment.

10. Initiate an initial ticket granting ticket for the principal using `kinit`. Go to the agent installation directory and run `kinit` and verify using:

```
klist: kinit
-k -t  {keytab file}  {principal name}
```

> **Note:**
>
> For auto renewal of Kerberos ticket, add the `hive.kerberos.principal.override` property in the `agent.properties` file as follows: `hive.kerberos.principal.override=<Principal name>`. If you do not mention the Principal name, then Oracle GoldenGate Veridata uses the principal name form the `database.url`.

11. Start the Oracle GoldenGate Veridata Agent.
12. Verify connection in UI.

> **Note:**
>
> In case you have permission-related issues, you may have to enter the Hadoop username and password in Database details. First, try without using the username and password.

7.5 Connecting Oracle GoldenGate Veridata to SSL-Enabled Oracle Database

SSL Enabled JDBC URL format: `jdbc:oracle:thin:@tcps://<host>:<port>/servicename?wallet_location=<wallet path>`, where `wallet_location` is the Directory path for client wallet files, which are `cwallet.sso` and `ewallet.p12`.

To connect Oracle GoldenGate Veridata to SSL-Enabled Oracle Database:

1. Create an auto-login wallet in the database as follows: `$ orapki wallet create -wallet <wallet path> -pwd <wallet password> -auto_login`.
2. Create a self-signed certificate and load it into the wallet: `$ orapki wallet add -wallet <wallet path> -pwd <wallet password> -dn "CN=<database hostname>" -keysize 1024 -self_signed -validity 3650`
3. Export the certificate so that you can load it into the client wallet: `$ orapki wallet export -wallet <wallet path> -pwd <wallet password> -dn "CN=<hostname>" -cert <server certificate path>`
4. Repeat step 1 to 3.
5. Exchange Client and Server Certificates:
    a. Load the server certificate into the client wallet: `$ orapki wallet add -wallet <client wallet path> -pwd <wallet password> -trusted_cert -cert <server certificate path>`

b. Load the client certificate into the server wallet: `$ orapki wallet add -wallet <server wallet path> -pwd wallet password -trusted_cert -cert <client certificate path>`

6. Check the contents of the server/client wallet: `$ orapki wallet display -wallet <server wallet path> -pwd <wallet password>`

For more information, see Create JKS Wallets for a TLS Connection to a DB System that has Client Authentication Enabled in the *Administering Oracle Data Safe guide*.

7.6 Connecting Oracle GoldenGate Veridata to SSL-Enabled MySQL Database

Setting up Server Authentication via server certificate

1. Copy `ca.pem` file from MySQL database server to veridata agent machine.
2. Run the `keytool` command in the veridata agent machine to import the `ca.pem` file: `$> keytool -importcert -alias <Set_Your_Alias> -file ca.pem -keystore truststore -storepass <Set_Your_Password>`

> **Note:**
>
> If the truststore file does not already exist, then a new one is created; else the certificate gets added to the existing file.

3. Append the following connection parameters to `MySQL JDBC URL` in the `agent.properties` file.
   - For MySQL version 8.0.12 and earlier: `database.url=jdbc:mysql://abc.com:3306?useSSL=true&verifyServerCertificate=true`.
   - For MySQL version 8.0.13 and later: `database.url=jdbc:mysql://abc.com:3306?sslMode=<VERIFY_CA or VERIFY_IDENTITY>`.
4. Export Java parameters to read the truststore you just created or modified. Export `JAVA_OPTS="$JAVA_OPTS -Djavax.net.ssl.trustStore=<path_to_truststore_file> -Djavax.net.ssl.trustStorePassword=<your_truststore_password>"`

Setting up Client Authentication via client certificate

1. Copy `client-cert.pem` and `client-key.pem` two files from MySQL database server to veridata agent machine.
2. Run the `openssl` command in the veridata agent machine to convert the client key and certificate files to a PKCS #12 archive: `$> openssl pkcs12 -export -in client-cert.pem -inkey client-key.pem -name "<Set_Your_name>" -passout pass:<Set_Your_Password> -out client-keystore.p12`
3. Run the `keytool` command in the veridata agent machine to import the PKCS file: `$> keytool -importkeystore -srckeystore client-keystore.p12 -srcstoretype pkcs12 -srcstorepass <Set_Your_Password> -destkeystore keystore -deststoretype JKS -deststorepass <Set_Your_Password>`.

> **Note:**
>
> If the keystore file does not already exist, then new one is created; else, the certificate is added to the existing file.

After the step, you can delete the PKCS #12 archive (client-keystore.p12 in the example).

4. Export java parameters to read the keystore you just created or modified: `export JAVA_OPTS="$JAVA_OPTS -Djavax.net.ssl.keyStore=<path_to_keystore_file> -Djavax.net.ssl.keyStorePassword=<your_keystore_password>" .`
5. Authentication via client certificate does not require connection parameters in `MySQL JDBC URL` as opposed to via server certificate.

2-Way Authentication

Apply the steps outlined in both *Setting up Server Authentication via server certificate* and *Setting up Client Authentication via client certificate* topics.

7.7 Connecting Oracle GoldenGate Veridata to SSL-Enabled SQL Server Database

SSL SYNTAX: `database.url=jdbc:sqlserver://<host>:<port>;databaseName=<dbName>;encrypt=false;trustServerCertificate=false;`

Example URL: `:database.url=jdbc:sqlserver://phoenix007.dev3sub2phx.databasede3phx.oraclevcn.com:1433;databaseName=atssrc;encrypt=false;trustServerCertificate=false;`

To connect Oracle GoldenGate Veridata to SSL-EnabledSQL Server:

1. Import the mssql crt file to the truststore: `keytool -importcert -alias ms_sql -keystore $JAVA_HOME/jre/lib/security/cacerts -storepass changeit -file <path>/mssql.pem`.
2. Update the jdbc url in `agent.properties`.
3. Set the path of truststore and truststore password in `$JAVA_OPTS` variable `export JAVA_OPTS="$JAVA_OPTS -Djavax.net.ssl.trustStore=<path_to_truststore_file> -Djavax.net.ssl.trustStorePassword=<your_truststore_password>"`
4. Start the agent.

7.8 Connecting Oracle GoldenGate Veridata to SSL-Enabled PostgreSQL Database

SSL SYNTAX: `database.url=jdbc:postgresql://<machine_name>:5432/<db_name>?` **`sslmode=verify-ca&sslrootcert=<full path of the cert>/root.crt`**

Example URL: `database.url=jdbc:postgresql://phoenix007:5432/postgres?` **`sslmode=verify-ca&sslrootcert=<path>/root.crt`**

To connect Oracle GoldenGate Veridata to SSL-Enabled PostgreSQL:

1. Copy the root certificate root.crt in your agent location and provide full path of it for `sslrootcert` parameters.
2. For `sslmode` there can be two values `verify-ca` and `verify-full`.

8
Migrate

8.1 Migrating Veridata Configuration from Veridata 12.2.1.4.0 to Oracle GoldenGate Veridata 23c

You cannot migrate users and user groups from Oracle GoldenGate Veridata 12c to 23c. You need to create users and user groups manually. It is a one-time task .The history will not be available in the new Oracle GoldenGate Veridata 23c post migration. You can view the historical information only from the previous version of Veridata 12c server.

To migrate the configurations, you need to use the Export utility in Oracle GoldenGate 12.2.1.4.0 and then import the xml by running the Import utility in Oracle GoldenGate Veridata 23c.

> **Note:**
>
> Prior to migration, ensure that you have installed the Oracle GoldenGateVeridata 23c binaries on a new server or in a new directory path of the existing server.

To migrate from Oracle GoldenGate Veridata 12c to 23c:

1. Go to the Oracle GoldenGate Veridata 12.2.1.4.0 installation location and run the Export utility. See Using the Veridata Import and Export Utilities in *Administering Oracle GoldenGate Veridata 12c*. An xml file gets created. You need to import this xml file by running the Import utility in Oracle GoldenGate Veridata 23c.
2. From the Oracle GoldenGate Veridata Web UI, run the Import utility. See Utilities. You can also run the Import utility as follows:

   ```
   ./vdt-import.sh -user <username> <-create| -delete | -update | -replace > <fileName>
   ```

   See Using the Import Utility.

9
Performance

9.1 Improving the Performance of Oracle GoldenGate Veridata

The following are some of the factors that influence the performance of Oracle GoldenGate Veridata and some ways you can improve its performance when processing large volumes of data.

Database and Network Use

Two critical performance factors for Oracle GoldenGate Veridata are:

- How data is sorted
- How the data is sent across the network

Performance Statistics for these performance factors are printed to the comparison report for each finished comparison and are recorded for the initial comparison step on the source and target. See Using the Comparison Report

Network Use

Oracle GoldenGate Veridata automatically optimizes its use of the network by using hashing and network message compression for efficient data transfer. The greater the size in bytes of the average row of a table or file (see the bytes/row performance statistic), the greater the rate of compression that is achieved through hashing (see the rh bytes/row and hash comp rate performance statistics). Whether a row is 50 bytes or 1000, the number of bytes that are used to represent its non-key values will be 12. Therefore, as a percentage of table size (in bytes), larger rows tend to use the network more efficiently. For those same reasons, the smaller the key size relative to row size, the more efficient the use of the network.

Additionally, on the NonStop platform check the send and receive TCP/IP buffer sizes. They should be set to 32K for Oracle GoldenGate Veridata.

Database Access

Database Sort: The following factors affect the performance of the database sorting mechanism:

- The number of rows in the tables being compared
- The indexes that are defined on the tables
- The keys that are being used
- The way that the database is tuned

After some test runs, if the performance of comparisons is not satisfactory, it might be faster to use server-side sorting (default sorting type), where Oracle GoldenGate Veridata Server itself performs the sorting.

Configuration Options

The following are some other things you can consider to improve comparison performance:

Partition Large Tables

You can divide large source and target tables into row partitions, each partition being associated with a different row subset. Row partitions enable you to process sets of data in parallel and also to control the timing of processing. For example, you can compare one partition today and the other one tomorrow. In addition, the results of a subset comparison can give you an idea of the synchronization status of the entire table.

Fetching COOS Rows as Batch from Database

To fetch the coos rows from the database either single or as batch rows, set `coos.batch.fetch=true` in `agent.properties`. By default, this property is set to `false`. In case of a large number (>10000) coos rows, then you can enable this property for better performance.

Fetching COOS Rows using Temporary Table

This is applicable only for Oracle Database. To optimize the performance of fetching of rows from database during the confirm-out-of-sync phase, specifically for table with no primary key or unique key, by default the coos strategy will be selected as coos join. For more information, see `coos.join.strategy` in Server Parameters for Confirm Out-of-Sync Fetch Strategy Parameters of the *Administering Oracle GoldenGate Veridata* guide. This strategy uses temporary table either Private or Global to fetch rows from compare pair table. The veridata user need to have create table privileges.

Exclude Columns

If a table contains columns that you know will never change, or if it does not matter whether those columns are in-sync, you can exclude those columns from the comparison to reduce the processing load. You exclude columns when you create or edit compare pairs.

Use delta processing

You can configure compare pairs to use delta processing, a performance feature whereby Oracle GoldenGate Veridata only compares data blocks that have changed, instead of comparing all of the rows in a table or file. For more information about delta processing, see Delta Processing.

Change the Database Transaction Isolation Level

Each Oracle GoldenGate Veridata agent has an agent.properties file in the root of its installation folder that contains environment parameters. One of those parameters is database.transaction.isolation. This property controls the transaction isolation level that is used during the initial comparison step. The default value for SQL Server and Teradata is `READ_UNCOMMITTED`. This means that the query that is issued to select rows can read rows that have been modified by other transactions but not yet committed (dirty reads). It does not issue shared locks to prevent other transactions from modifying the data, nor is it blocked by the locks of other transactions.

The advantage of using the `READ UNCOMMITTED` option is that the initial reads of the data are faster because they are not affected by locking. The negative impact of this is that more records could be labeled as possibly out-of-sync (because the data can change over the course of the transaction) and would need to be compared again during the confirmation step. If you think that there are too many rows being compared in the confirmation step, you can edit

the properties file and set database.transaction.isolation to `COMMITED`, which only permits the fetching of committed records. You must weigh any improvement against the possibility that the initial comparison step becomes slower due to the affect of locks to preserve read consistency.

> **Note:**
>
> (The only value that is supported for Oracle is `READ_COMMITTED`, and the confirmation step always uses READ_COMMITTED, because at this stage dirty reads are not acceptable.)

Profile options

You can control certain parameters for the initial and confirmation steps that can help increase performance.

Initial comparison step parameters

- **Limit the number of rows that are compared**: By using the Limit Number of Input Rows parameter for a specific job profile, you can constrain the number of rows that are fetched for processing. This enables you to process a smaller number of rows to get an idea of how out-of-sync (or not) the entire table is. Based on the results, you can make a decision about whether to run a complete comparison or just resynchronize the data. The Limit Number of Input Rows parameter is a General parameter for the initial comparison process.
- **Increase process priority (NonStop only)**: Assign the Oracle GoldenGate Veridata Agent the highest process priority possible on a NonStop system. You can assign a priority (as well as a process name and CPU number) by using the NonStop settings of the initial and confirmation steps in the job profile.
- **Increase processing threads**: The default is four threads. If the machine where Oracle GoldenGate Veridata Server is running has more processors, you can change the value of the Max Concurrent Comparison Threads parameter accordingly, keeping all threads busy by executing simultaneous comparisons. This parameter is in the General profile settings of the initial comparison process.

Confirmation step parameters

- **Run each comparison step separately**: By default, Oracle GoldenGate Veridata runs the initial compare and confirmation processes concurrently. If you run them in sequence, fewer system resources are used, but it will take longer to get results. This functionality is controlled by the Run Concurrently with Initial Compare parameter of the comparison step's General profile settings.
- **Skip the confirmation step**: The default is to always perform a confirmation step. You can skip this step if the database is quiesced or if replication is not actively replicating data changes. Use the Perform Confirmation Step parameter of the confirmation step's General profile settings.
- **Increase process priority (NSK)**: Assign the Oracle GoldenGate Veridata Agent the highest process priority possible on a NonStop system. You can assign a priority (as well as a process name and CPU number) by using the NonStop settings of the initial and confirmation steps in the job profile.

Connection options

Try increasing the batch size of fetched rows to increase throughput. To do this, increase the size of the Initial Compare Fetch Batch Size parameter for the initial and confirmation steps.

9.2 Performance Statistics

The two most critical aspects of Oracle GoldenGate Veridata performance are database access and network usage. Performance statistics for both of these aspects are printed to the comparison report for each comparison performed and are recorded for the initial comparison step on the source and target systems. The following describes these statistics. Depending on the results of these statistics, there are ways to optimize database access and network usage.

duration

The time spent processing the fetched rows.

rows fetched

The number of rows fetched from the database.

rows/sec

The number of rows processed per second.

bytes fetched

The total number of bytes that were processed.

bytes/sec

The number of rows, in terms of bytes, processed per second.

lob chunks fetched

The number of 32k blocks of LOB data fetched.

batches fetched

The number row batches fetched. The default is 10 rows per batch.

ipc msgs

The number of interprocess messages between the server and agent processes.

ipc bytes

The number of bytes transferred between the server and agent processes.

bytes applied

The number of bytes per message that were applied.

lob chunks applied

The number 32k byte LOB chunks applied at the target database.

lob fetch time duration (secs)

The amount of time spent fetching LOB data.

batches applied

The total number of bytes that were processed.

transaction batches

The number of transactions used to apply the data.

transaction single statements

the number of single row transactions applied during error recovery.

10

Accessibility

10.1 About this Accessibility Article

This article consists of information about accessing Oracle GoldenGate Veridata with a assistive technologies and a keyboard, and guidelines for developing Oracle GoldenGate Veridata applications that are accessible and highly usable to all users.

"*Oracle is committed to creating accessible technologies and products that enhance the overall workplace environment and contribute to the productivity of our employees, our customers, and our customers' customers.*" —Safra Catz, Chief Executive Officer, Oracle.

10.1.1 Accessibility Overview

Web accessibility means people with disabilities can effectively use and contribute to the web. To ensure people with disabilities can use Oracle GoldenGate Veridata, we need to design and develop more accessible web applications and software.

Accessibility is not just a checklist, but an ongoing and planned effort. The aim of this guide is to help you start the planning process, and understand how to build accessible applications with Oracle GoldenGate Veridata.

10.1.2 What is Web Accessibility?

This article refers to the Web Content Accessibility Guidelines 2.0 (WCAG 2.0) to back up some of the guidelines with the relevant standard and provide further reading.

According to W3C, web accessibility means that people with disabilities can perceive, understand, navigate, and interact with the web, and that they can contribute to the web. The disabilities that affect access to the web can be categorized as: visual, auditory, speech, cognitive, physical, and neurological. The goal of web accessibility is to provide equal access to users with disabilities. That is, developers need to focus on building more accessible web applications and products. Apart from people with disabilities, the elderly with increasing impairments, and people in a limiting situation (for example, slow internet connection, or no audio) can also benefit with accessible applications and products.

Principles of Accessibility

WCAG 2.0, which is created by W3C defines the four principles of accessibility as:

- **Perceivable**: Information and user interface components must be presentable to users in ways they can perceive. That is, users must be able to perceive the information presented.
- **Operable**: User interface components and navigation must be operable. That is, users must be able to operate the interface.
- **Understandable**: Users must be able to understand the information as well as the operation of the user interface.
- **Robust**: Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies.

Each principle includes a list of guidelines that address the principle and there are a total of 12 guidelines. Each guideline includes one or more testable success criteria, which are at three levels: A, AA, and AAA. To learn more about these principles, guidelines, and success criteria, see Web Content Accessibility Guidelines 2.0 (WCAG 2.0).

10.1.3 Why Accessibility is Important?

Up to 16% of the world population is disabled, either through birth, ageing, illness, or the result of an accident. Accessible products remove obstacles between a company and its existing or potential customers. Accessibility enables employers to recruit from a broader pool of talent. Overall, accessibility generally improves product for everyone (Universal Design).

Developing accessible products is very important for companies in order to adhere to Procurement and Discrimination laws such as:

- Section 508 of the Rehabilitation Act of 1973
- Mandate 376 (E.U.)
- Accessibility for Ontarians with Disabilities Act (Canada)
- Americans with Disabilities Act (U.S.)
- Disability Discrimination Act, and Equality Act (U.K.)
- Twenty-First Century Communications and Video Accessibility Act (U.S.)

.

10.1.4 About Building for Accessibility

Building for accessibility means coding to standards and not for a specific technology. Building to standards often leads to better, standardized HTML and cleaner code. The process of building accessible applications at Oracle involves considerable amount of time and resources at all stages of the development cycle. Accessibility guidelines and standards are considered, implemented, and reviewed through the Voluntary Product Accessibility Template (VPAT, a product's statement of conformance to Section 508 of the U.S. Federal Rehabilitation Act).

The Oracle Accessibility Program Office, reporting to the office of the Chief Corporate Architect, is responsible for defining the corporate standards for accessibility, and developing materials to train all employees so that they can successfully create products that meet those standards. To learn more about Oracle's Accessibility Program, see Oracle Accessibility Information and Resources.

Oracle uses the VPAT to represent the degree of conformance to various accessibility standards and guidelines, including Section 508, WCAG 1.0, and WCAG 2.0. Depending on when a product was developed and released, different standards may be listed.

.

10.2 Using Oracle GoldenGate Veridata with Assistive Technologies and Keyboard

You can navigate and perform user interface (UI) actions without a mouse by using keyboard controls. Common keyboard controls are available in all applications. Additional keyboard controls are used for complex interactions and vary by component type.

Table 10-1 Frequently used keyboard controls

Key(s)	Action
Enter	Trigger the activity, when the focus is on a link, an image, or a button with an associated URL or activity. Toggle the focus or selection. For input fields and menus, submit or enter a typed value or select the highlighted item. In Tables, make the current row editable or enter Actionable Mode.
Shift + Enter	For cells in editable Tables, make the previous row editable and move the focus to the editable cell in the current column of the previous row.If the first row is editable, make it read-only.
Space	Trigger the activity, when the focus is on a link, an image, or a button with an associated URL or activity. Toggle the focus or selection.
Shift + Space	In components such as Card View, Data Grid, List View, and Tree View, selects contiguous items from the last selection to the current item.
Ctrl / Cmd + Space	Multi-select item with focus. In List View and Tree View, toggles the selection of the current item while maintaining previous selections.
Tab	Sets focus to the first focusable element or cycles focus through next focusable elements. For some components, such as Card View, Data Grid, List View, Table, or Waterfall Layout, when in Tabbable or Actionable Mode, if the focus is on the last element, the focus goes back to the first focusable element. When not in Tabbable or Actionable mode, the focus goes to the next focusable item on the page.
Shift + Tab	Cycles focus through previous elements. For some components such as Card View, Data Grid, List View, Table, or Waterfall Layout, when in Tabbable or Actionable Mode, if the focus is on the first element, the focus goes back to the last focusable element. When not in Tabbable or Actionable mode, the focus goes to the previous focusable item on the page.

Table 10-1 (Cont.) Frequently used keyboard controls

Key(s)	Action
Up Arrow	Moves focus and selection to previous item or scrolls the content up. For drop down lists, expands the list when closed. For numerical input fields, increases the number by one in the segment. For slider components, scrolls right on horizontal sliders and up on vertical sliders.
Down Arrow	Moves focus and selection to next item or scrolls the content down. For drop down lists, opens the list when closed. For numerical input fields, decreases the number by one in the segment. For slider components, scrolls left on horizontal sliders and down on vertical sliders.
Left Arrow	Moves focus and selection to the previous item or the item on the left. Can also scroll the content left. For collapsible groups inside collection components collapses the current expanded item. For slider components, scrolls left on horizontal sliders and down on vertical sliders.
Right Arrow	Moves focus and selection to the next item or the item on the right. Can also scroll the content right. For collapsible groups inside collection components expands the current item if it has children. For slider components, scrolls right on horizontal sliders and up on vertical sliders.
Shift + Arrow	For selectable elements within collection components, moves focus and extends the selection in the direction of the arrow
Shift + Ctrl / Cmd + Arrow	In Card View, reorders the current item in the direction of the arrow.
F2	For some components, such as Cards, Data Grid, List View, Table, or Waterfall Layout, enters Tabbable or Actionable Mode allowing navigation and use of controls in the item. Exits Tabbable or Actionable Mode.
Alt + F10	For Rich text input, moves focus to the text editing toolbar.
Esc	Exits Tabbable or Actionable Mode. Collapses drop down lists.
= or +	Zoom in one level (if zooming is enabled).
- or _	Zoom out one level (if zooming is enabled).
Page Up	For Data grid and data visualization components, such as Diagram and Timeline, pans up (if scrolling is enabled).

Table 10-1 (Cont.) Frequently used keyboard controls

Key(s)	Action
Page Down	For Data grid and data visualization components, such as Diagram and Timeline, pans down (if scrolling is enabled).
Shift + Page Up	For Data grid and data visualization components, such as Diagram and Timeline, pans left (in left-to-right locales). Pans right (in right-to-left locales).
Shift + Page Down	For Data grid and data visualization components, such as Diagram and Timeline, pans right (in left-to-right locales). Pans left (in right-to-left locales).
Home or End	For Data grid, moves focus to the first or last focusable item.
Ctrl / Cmd + X	For draggable elements, marks the selected items to move if reorder is enabled.
Ctrl / Cmd + C	For draggable elements, marks the selected items to copy if reorder is enabled.
Ctrl / Cmd + V	For draggable elements, pastes selected items directly before the current item (or as the last item if the current item is a folder).

Note:

For more information about Oracle's commitment to accessibility, see Oracle's Accessibility Program.

- Component-Specific Keyboard Controls

10.2.1 Component-Specific Keyboard Controls

Some components use additional or alternative keyboard controls. For more details, review the specific component information on the Oracle JavaScript Extension Toolkit (JET) Keyboard and Touch Reference or navigate the links listed in the following table.

Table 10-2 Component-Specific Keyboard Controls

Collections	Controls	Forms and Inputs	Layouts and Navigation	Visualizations
Data grid	Button	Checkbox	Accordion	Area chart
Indexer	File picker	Checkbox set	Action card	Chart
List view	Film strip	Color palette	Collapsible	Diagram
Row expander	Menu	Color spectrum	Conveyor belt	Gantt
Stream list	Menu button	Date picker	Dialog	Legend
Swipe actions	Message	Input date	Drawer layout	Line chart
Table	Message banner	Input date mask	Navigation list	Meter bar
Tree view	Message toast	Input date text	Popup	Meter circle
Waterfall layout	Train	Input date time	Tab bar	Nbox
		Input number		Picto chart
		Input password		Rating gauge
		Input search		Sunburst
		Input sensitive text		Tag cloud
		Input text		Thematic map
		Input time		Timeline
		Radioset		Treemap
		Select multiple item		
		Select single item		
		Slider		
		Range slider		
		Switch		

Note:

For more information about Oracle's commitment to accessibility, see Oracle's Accessibility Program.