

Oracle® GoldenGate

Administering Oracle GoldenGate Veridata



12c (12.2.1.4)

F16425-15

October 2023

The Oracle logo, consisting of the word "ORACLE" in white, uppercase letters, centered within a solid red square.

ORACLE®

Copyright © 2015, 2023, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

Audience	ix
Documentation Accessibility	ix
Related Information	ix
Conventions	ix

1 Introduction to Oracle GoldenGate Veridata

1.1	Oracle GoldenGate Veridata Architecture	1-1
1.2	Configuring Single Sign-on for Oracle GoldenGate Veridata	1-3
1.3	Comparing Data with Oracle GoldenGate Veridata	1-3
1.3.1	Oracle GoldenGate Veridata Comparison Objects	1-4
1.3.1.1	Supported Datatypes - Oracle	1-4
1.3.1.2	Supported Datatypes - SQL Server	1-5
1.3.1.3	Supported Datatypes - PostgreSQL	1-6
1.3.1.4	Supported Datatypes - MySQL, MariaDB	1-8
1.3.1.5	Supported Datatypes - Teradata	1-10
1.3.1.6	Supported Datatypes - DB2 LUW	1-11
1.3.1.7	Supported Datatypes - DB2 for i	1-12
1.3.1.8	Supported Datatypes - DB2 z/OS	1-12
1.3.1.9	Supported Datatypes - Sybase	1-13
1.3.1.10	Supported Datatypes - Informix	1-14
1.3.1.11	Generic Limitations and Clarifications	1-15
1.3.2	Satisfying Uniqueness Requirements	1-15
1.3.3	How Oracle GoldenGate Veridata Compares Data	1-16
1.3.3.1	Initial Comparison Step	1-16
1.3.3.2	Confirmation Step	1-17
1.4	Viewing Comparison Results	1-17

2 Configuring Security

2.1	Overview of Oracle GoldenGate Veridata Security	2-1
-----	---	-----

2.2	Configuring SSL Connection between Oracle GoldenGate Veridata Server and Agents	2-1
2.2.1	One-Way SSL Connection	2-1
2.2.2	Two-Way SSL Connection	2-2
2.2.3	Generating Self-Signed Keystores and Truststores	2-3
2.2.4	Configuring SSL Settings for the Oracle GoldenGate Veridata Agent	2-6
2.2.5	Updating Oracle GoldenGate Veridata Agent Wallet	2-7
2.2.6	Configuring SSL Settings for the Oracle GoldenGate Veridata Server	2-7
2.2.7	Configuring an SSL Connection between Oracle GoldenGate Veridata Server and NonStop Agent	2-8
2.2.8	Updating Oracle GoldenGate Veridata Server Trust Store	2-8
2.3	Securing the Oracle GoldenGate Veridata Files	2-9
2.4	Securing Access to Oracle GoldenGate Veridata by Defining User Groups	2-10
2.5	Creating and Editing Users	2-14
2.6	Changing Database Schema Passwords	2-15
2.7	Encrypting Report Files	2-16
2.7.1	Enabling Report Encryption	2-17
2.7.2	Displaying Encrypted Files with the reportutil Utility	2-17

3 Managing C-Agent, Manager, and Java Components

3.1	Starting and Stopping the C-Agent and the Oracle GoldenGate Veridata Manager	3-1
3.2	Starting and Stopping the Java-Based Components	3-1
3.3	Reloading Logging Information	3-2
3.4	Controlling Logging Levels for Oracle GoldenGate Veridata Agent	3-2
3.5	Connecting to the Oracle GoldenGate Veridata Web Interface	3-3

4 Managing Identities and Comparing Data

4.1	Overview of the Vericom Tool	4-1
4.2	Running the Vericom Tool	4-1
4.2.1	SSL Configuration for Vericom	4-10
4.3	Managing Identities in a Credential Store	4-11
4.3.1	Adding a Credential Store	4-12
4.3.2	Creating an Alias	4-12
4.3.3	Deleting a Credential Store	4-12
4.3.4	Using the Alias	4-12
4.3.5	Display Alias	4-13
4.3.6	Updating the Alias	4-13
4.3.7	Deleting the Alias	4-13
4.4	Vericom Exit Statuses	4-13

5 Using the Veridata Import and Export Utilities

5.1	Introduction to the Import and Export Utilities	5-1
5.1.1	Supported Configurations	5-1
5.2	Running the Import and Export Utilities	5-1
5.2.1	Using the Export Utility	5-2
5.2.2	Using the Import Utility	5-3
5.2.3	Processing the Configuration	5-3
5.2.4	SSL Configuration for Export and Import Utilities	5-5
5.3	Configuration File Element Reference	5-6
5.4	configuration	5-7
5.5	column	5-8
5.6	colfilter	5-9
5.7	colfiltercol	5-10
5.8	compare-pair	5-10
5.9	connection	5-13
5.10	conn-properties	5-15
5.11	delta-config	5-15
5.12	description	5-16
5.13	enscribe-info	5-16
5.14	enscribe-key	5-17
5.15	excluded-column	5-18
5.16	expandddl	5-18
5.17	filter	5-19
5.18	group	5-20
5.19	job	5-22
5.20	profile	5-23
5.21	key-column	5-24
5.22	profile-general	5-24
5.23	sorting-method	5-24
5.24	initial-compare	5-25
5.25	confirm-out-of-sync	5-25
5.26	param	5-25
5.27	repair	5-26
5.28	sql-partition	5-26
5.29	table partition	5-27

6 Running Veridata GoldenGate Parameter Processing

6.1	Overview of the Command-Line Interface	6-1
6.2	Running the Veridata GoldenGate Parameter Processing	6-1
6.2.1	Using a Property File	6-3
6.3	Parameter Handling	6-4
6.4	Map and Table Statement Handling	6-5
6.5	VGPP Example	6-7

7 Oracle GoldenGate Veridata Server Configuration Parameters

7.1	Overview of the Server Memory	7-1
7.2	Estimating Memory Usage	7-1
7.3	How to Set a Parameter	7-1
7.4	Parameter Descriptions	7-1
7.4.1	Server Parameters	7-1
7.4.1.1	database.hash	7-2
7.4.1.2	max_lob_key_len	7-2
7.4.1.3	server.concurrent.readers	7-2
7.4.1.4	server.concurrent.writers	7-2
7.4.1.5	server.convert_empty_to_null	7-3
7.4.1.6	server.mapped_sort_buffers	7-3
7.4.1.7	server.max_concurrent_comparison_threads	7-4
7.4.1.8	server.max_concurrent_jobs	7-4
7.4.1.9	server.max_sort_memory	7-4
7.4.1.10	server.memory_mapped_sort_directory	7-5
7.4.1.11	server.meta_session_handle_timeout	7-5
7.4.1.12	server.persistence_db_type	7-5
7.4.1.13	server.veridata_data	7-5
7.4.1.14	socket.timeout.mins	7-6
7.4.1.15	truncate_spaces_len	7-6
7.4.2	Parameters for Configuring SSL Communication	7-6
7.4.2.1	server.useSsl	7-7
7.4.2.2	server.ssl.client.allowTrustedExpiredCertificates	7-7
7.4.2.3	server.ssl.client.identitystore.keyfactory.alg.name	7-7
7.4.2.4	server.ssl.client.truststore.keyfactory.alg.name	7-8
7.4.2.5	server.ssl.algorithm.name	7-8
7.4.3	Parameters for Veridata Command-Line Utility	7-8
7.4.3.1	veridata.cli.run_from_managed_server	7-8
7.4.3.2	veridata.cli.server.listenAddress	7-9
7.4.3.3	veridata.cli.managed_server_name	7-9
7.4.3.4	veridata.cli.server.timeout.seconds	7-9

7.4.4	Parameters for Report File Encryption	7-9
7.4.4.1	server.encryption	7-10
7.4.4.2	server.encryption.bits	7-10
7.4.5	Parameters for Email Notifications and Alerts	7-10
7.4.5.1	mail.enabled	7-10
7.4.5.2	mail.alert.only	7-11
7.4.5.3	mail.smtp.server	7-11
7.4.5.4	mail.smtp.port	7-11
7.4.5.5	mail.smtp.connection.ssl.tls	7-11
7.4.5.6	mail.from	7-12
7.4.5.7	mail.to	7-12
7.4.6	Server Parameters for Oracle Streams Tagging	7-12
7.4.6.1	repair.oracle.tag.enable	7-12
7.4.6.2	repair.oracle.tag	7-12

8 Agent Parameters

8.1	compare.xmldatatype.format	8-1
8.2	network.checksum.level	8-1
8.3	network.checksum.types	8-1
8.4	network.encryption.level	8-1
8.5	network.encryption.types	8-2
8.6	server.number_sort_threads	8-2
8.7	rowscn	8-2

A Moving from a Test to Production Environment

A.1	Moving Installations from a Source Environment to a Target Environment	A-1
A.2	Additional Steps for Moving Oracle GoldenGate Veridata Repository	A-1
A.2.1	Moving Veridata Configuration Data from Test to Production	A-1
A.2.2	Applying Configuration Changes while Moving from Test to Production	A-2
A.2.3	Modifying the Agent details in the Production Environment	A-3

B Sample Configuration File

B.1	Sample Configuration File	B-1
-----	---------------------------	-----

C Profile Parameters

C.1	General (profile-general)	C-1
C.2	Sorting Method (sorting-method)	C-1
C.3	Initial Compare (initial-compare)	C-2

C.4	Confirm-Out-Of-Sync (confirm-out-of-sync)
C.5	Repair (repair)

C-3
C-5

Preface

This document describes how to configure and administer Oracle GoldenGate Veridata.

Audience

This document is intended for installers and system administrators who are installing, configuring and running Oracle GoldenGate Veridata.

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

The other Oracle GoldenGate Veridata documents for this release are as follows:

- Upgrading Oracle GoldenGate Veridata
- Using Oracle GoldenGate Veridata
- Oracle GoldenGate Veridata Release Notes
- Installing Oracle GoldenGate Veridata

The Oracle GoldenGate Product Documentation Libraries are found at

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

For additional information on Oracle GoldenGate, refer to, <https://www.oracle.com/middleware/technologies/goldengate.html>

Conventions

The following text conventions are used in this document:

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

1

Introduction to Oracle GoldenGate Veridata

Oracle GoldenGate Veridata compares one set of data to another and identifies data that is out-of-sync, and allows you to repair that data.

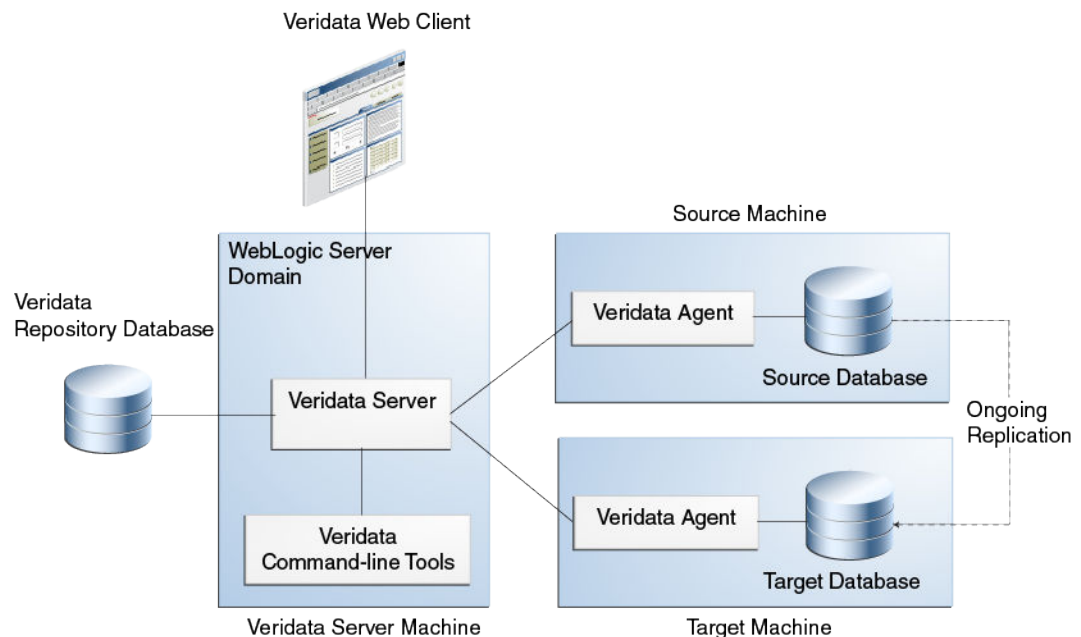
This chapter includes the following sections:

1.1 Oracle GoldenGate Veridata Architecture

Oracle GoldenGate Veridata compares one set of data to another, identifies data that is out-of-sync, and enables you to repair any out-of-sync data. Oracle GoldenGate Veridata supports high-volume, 24x7 heterogeneous replication environments where downtime to compare data sets is not an option. By accounting for data that is being replicated while a comparison takes place, Oracle GoldenGate Veridata can run concurrently with data transactions and replication and still produce an accurate comparison report.

Oracle GoldenGate Veridata maps column data types across different types of databases automatically. For more information about this feature in Oracle GoldenGate Veridata web user interface, see the online help. Alternatively, you can map columns manually if automatic mapping does not accommodate format differences in a heterogeneous environment. To manually upload an XML file, use the `veridata_import` utility. For more information, see [Using the Veridata Import and Export Utilities](#).

Figure 1-1 Oracle GoldenGate Veridata Architecture



Oracle GoldenGate Veridata Server

The Oracle GoldenGate Veridata Server performs the following functions:

- Coordinate the execution of Oracle GoldenGate Veridata tasks
- Sort rows
- Compare data
- Confirm out-of-sync data
- Produce a report for review

Oracle GoldenGate Veridata Web User Interface

The Oracle GoldenGate Veridata web user interface (UI) is a browser-based graphical user interface for these activities:

- Configure comparison objects and rules
- Initiate comparisons
- Review the status and output of comparisons
- Repair out-of-sync data
- Review out-of-sync data

Oracle GoldenGate Veridata Repository

The Oracle GoldenGate Veridata repository is a collection of database objects that persists configuration information to disk, saving it permanently as a user environment.



Note:

Out-of-sync data is not stored in the repository. This data is stored on the server's file system.

Oracle GoldenGate Veridata Agent

The Oracle GoldenGate Veridata Agent (agent) executes the following database-related requests on behalf of the Oracle GoldenGate Veridata server:

- Hash rows for initial comparison
- Fetch and update rows to repair out-of-sync data
- Return column-level details for out-of-sync rows

Oracle GoldenGate Veridata Manager

The Oracle GoldenGate Veridata Manager is part of the C-code agent that is required for the NonStop platform. It controls the agent process.

The Manager is not used in a Java agent, which is used for the other databases that are supported by Oracle GoldenGate Veridata.

Oracle GoldenGate Veridata Command Line Utilities

Oracle GoldenGate Veridata includes the following command-line utilities:

Table 1-1 Command-Line Utilities

Name	Description
vericom	Enables you to run comparisons by using automated programs. See Managing Identities and Comparing Data .
veridata_import	Maps comparison objects and rules in an XML file and imports it into the repository. See Using the Veridata Import and Export Utilities .
veridata_export	Maps comparison objects and rules in the repository and exports them to an XML file. See Using the Veridata Import and Export Utilities .
veridata_param_process	Helps to use the Oracle GoldenGate parameter files. See Introduction to Oracle GoldenGate Veridata .
reportutil	Supports viewing encrypted report files and out-of-sync data.

1.2 Configuring Single Sign-on for Oracle GoldenGate Veridata

Oracle GoldenGate Veridata 12c (12.2.1) supports single sign-on (SSO) for authentication. When you configure SSO, set the SSO properties of the server and configure the sign-out URL for the SSO session.

To configure SSO for Oracle GoldenGate Veridata server:

- **Run the `configureVeridata` script:** `DOMAIN_HOME/veridata/bin/configureVeridata.sh -pUweb.singleSignOutUrl=Single sign out URL.`

If your domain no longer uses SSO, then remove the SSO logout configuration:

```
DOMAIN_HOME/veridata/bin/configureVeridata.sh -
pUweb.singleSignOutUrl=default.
```

The parameter usage is explained in the `veridata.cfg` file:

```
# (web.singleSignOutUrl) as
# web.singleSignOutUrl - Specify the Single Sign Out URL here:
# Formats: /oamssso/logout.html?end_url=/veridata
# http://myoamserverhost:port/oam/server/logout?end_url=http://
my.veridata.site.com:veridata-port/veridata
# http://myoamserverhost:port/oamssso/logout.html?end_url=http://
my.veridata.site.com:veridata-port/veridata

# This URL must conform to the grammar in RFC 2396, except the few deviations
mentioned in the java documentation for construction of a URI by parsing the given
string.

web.singleSignOutUrl default
```

1.3 Comparing Data with Oracle GoldenGate Veridata

This section explains how to configure the objects that are to be compared and how Oracle GoldenGate Veridata processes comparisons.

1.3.1 Oracle GoldenGate Veridata Comparison Objects

To begin using Oracle GoldenGate Veridata, you create the following objects that help you manage your work and identify the data that you want to compare:

- Data source connections
- Groups
- Compare Pairs
- Profiles
- Jobs

To order rows for comparison, Oracle GoldenGate Veridata relies on a unique identifier (primary key, unique key, user-defined key).

1.3.1.1 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
- XML

- INTERVAL YEAR [(year_precision)] TO MONTH
- INTERVAL DAY [(day_precision)] TO SECOND [(fractional_seconds_precision)]
- FLOAT
- UDTs

Limitations and Clarifications:

Both Object types and Collection types UDTs are supported

1.3.1.2 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
- DATETIMEOFFSET
- 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*.

1.3.1.3 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_Isn
- pseduo types
- Point
- path
- polygon
- range types
- tsvector
- tsquery
- enum
- domain

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 formatxx: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 --> 00:01:38Therefore, 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.

1.3.1.4 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

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 be 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.

1.3.1.5 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

1.3.1.6 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

1.3.1.7 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.

1.3.1.8 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

1.3.1.9 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

1.3.1.10 Supported Datatypes - Informix

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

- bigint
- bigserial
- blob
- boolean
- byte
- char
- clob

- date
- datetime hour to second
- datetime year to day
- datetime year to fraction(5)
- datetime year to second
- decimal
- float
- html
- int8
- integer
- interval
- lvarchar
- money
- nchar
- nvarchar
- serial
- serial8
- smallfloat
- smallint
- text
- varchar

1.3.1.11 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.

1.3.2 Satisfying Uniqueness Requirements

Oracle GoldenGate Veridata relies on a unique identifier to order rows for comparison.

- **Primary Key:** By default, Oracle GoldenGate Veridata uses the primary key if one is available.

- **Unique Key:** If no primary key is defined, then Oracle GoldenGate Veridata uses the smallest unique index.
- **User-defined Key:** If a table doesn't have a primary or a unique key, then all columns are mapped for comparison. Later, you can define an existing index or a set of columns for comparison purposes when you define a compare pair. However, although primary or unique keys can be mapped automatically, user-defined keys must be mapped manually. If you prefer a different ordering method, then you can also use a user-defined key to override existing keys or indexes.

1.3.3 How Oracle GoldenGate Veridata Compares Data

Comparison activities consist of an initial comparison step and a confirmation step. To change the steps' aspects, change the parameters in the Oracle GoldenGate Veridata web user interface.

1.3.3.1 Initial Comparison Step

In the *initial comparison* (or *row hash*) step, rows are retrieved from the source and target tables with a query. If the source and target databases are of different types, the columns are converted to a standardized data type format for accurate comparison. By default, Oracle GoldenGate Veridata compares rows by comparing all columns of the primary key literally (value-for-value) and by using a hash value for all non-key columns. The unique digital signature that is used to calculate the hash value shrinks the data to be transferred over the network for the comparison. The signature still provides a highly reliable (but not absolute) and efficient mechanism for determining whether two rows contain the same or different column values.

To ensure that you discover out-of-sync rows, you can configure Oracle GoldenGate Veridata to compare non-key rows column-for-column. Full-column comparisons reduce the processing performance in proportion to the number of columns, and they increase network usage.

For all supported databases, you can use the delta processing performance feature if you are using server-side sorting. For this processing, Oracle GoldenGate Veridata detects which data blocks were modified since a previous comparison and only compares the rows in those blocks. Rows in unchanged blocks are skipped. The default is to compare all rows.

Delta processing consists of two steps:

- Collect the base modification time of the previous run for subsequent delta comparisons. This step is always included when you enable delta processing for a compare pair.
- Compare data that was modified since the base comparison and use the information that was collected in the first step. To enable this step, click **Enable Delta Processing** on the Compare Pair Configuration page and the Run/Execute Job page in the Oracle GoldenGate Veridata web user interface. Disable delta comparison when modifications, such as table reorganizations, invalidate the collected delta base information.

After the initial comparison, rows that appear to be out-of-sync are stored in a maybe out-of-sync (MOOS) queue in memory, because the comparison is inconclusive. When a replication is working concurrently with a comparison, especially if there is replication

latency, rows can appear to be out-of-sync. However, the current data is in flight (somewhere in the replication flow), and replication resynchronizes them.

1.3.3.2 Confirmation Step

The *confirmation*, or *confirm-out-of-sync (COOS)*, step ensures accurate results by confirming the row status in a changing environment. This step involves predicated queries on the source or target database by using the rows extracted from the MOOS queue. The status is evaluated as one of the following:

- *in-flight*: The row was out-of-sync in the initial comparison step, but it was updated. It is assumed that replication or another mechanism applied the change, but Oracle GoldenGate Veridata was unable to confirm that the rows were in-sync.
- *in-sync*: The source row values were applied to the target row by replication or another method. An in-sync status does not guarantee that the rows are synchronized at any particular moment if the underlying tables are continuously changing, but it does indicate that replication is working.
- *persistently out-of-sync*: The row was not updated since the initial comparison step, and can be assumed to be out-of-sync.

By default, confirmation processing occurs in a thread that is parallel to the initial comparison step, but the confirmation of each row waits until a specified replication latency threshold expires. For example, if latency is 60 seconds, and the initial comparison step revealed an out-of-sync row at 9:30, then the confirmation step for that row is not performed until 9:31 to allow replication to apply any change that was in-flight. After latency is accounted for, rows can be confirmed as persistently out-of-sync and are stored in one or more out-of-sync reports.

1.4 Viewing Comparison Results

When a job is completed, you can view the out-of-sync report and the comparison reports by using the Oracle GoldenGate Veridata web user interface or by viewing the files themselves.

If report encryption is enabled for the server, then use the reportutil tool to view the report files. See [Encrypting Report Files](#). The Oracle GoldenGate Veridata web user interface automatically decrypts the file before displaying them.

You can store an OOS report in binary format, in XML format, or both (or none).

- **OOS file**: When stored in binary form, the OOS report contains out-of-sync comparison results that you use for viewing row differences in the Oracle GoldenGate Veridata web user interface. You can also use the report to recompare out-of-sync rows later. To recompare rows, select run options to execute another confirmation step. The step compares the current state of just those rows and then reports which rows remain out-of-sync after replication or another restorative procedure was applied.
- **OOSXML file**: When the OOS report is stored as XML, it is written to an OOSXML file and is stored to conform to an internal XML schema. Among the many advantages of XML, the most significant is that it can be easily manipulated by many tools. In its XML form, the file contains all information, including metadata, that is needed to select rows for resynchronization by external programs.

Each finished job, group, and compare pair generates a comparison report with the following type of information:

- Comparison parameters used

- Number of rows compared and out-of-sync
- Timing of the comparison
- Performance statistics
- Source and target data values

By default, the OOS files are located in the following subdirectories of the server installation directory. You can change the default location by specifying another path for the `server.veridata_data` property in the `veridata.cfg` file:

- **OOS files:** `VERIDATA_DOMAIN_HOME/veridata/reports/oos`
- **OOSXML files:** `VERIDATA_DOMAIN_HOME/veridata/reports/oosxml`

These subdirectories are further organized by run ID, job name, group name, and compare pair. In the OOSXML directory, the `.oosxml` files are the control files. The files with sequential file extensions are the JSON chunks. The JSON data is spread into multiple files (called "chunks") for performance purposes. You can encrypt the comparison reports. For more information, see [Encrypting Report Files](#).

2

Configuring Security

Oracle GoldenGate Veridata provides a safe, secure environment for your business data by using Secure Sockets Layer (SSL) and plain socket communications. You can control your security by managing passwords and encrypting the report files.

This chapter includes the following topics:

2.1 Overview of Oracle GoldenGate Veridata Security

When using Oracle GoldenGate Veridata you select, view, and store data values from the tables or files of your business applications. Ensure to protect access to the following components:

- Files, programs, and directories in the Oracle GoldenGate Veridata installation directories.
- Data files that contain the results of data comparisons
- Oracle GoldenGate Veridata web user interface, where data values can be viewed

2.2 Configuring SSL Connection between Oracle GoldenGate Veridata Server and Agents

Oracle GoldenGate Veridata supports SSL communication between Oracle GoldenGate Veridata Server and multiple Oracle GoldenGate Veridata Agents that are connected over a network. This section describes how to configure SSL between the Oracle GoldenGate Veridata server and the agents.

In SSL scenario, the Oracle GoldenGate Veridata server is considered the SSL Client and the agents are considered the SSL Servers. The server and the agents authenticate each other's identity. The data exchanged between the server and agent is also encrypted.

Oracle GoldenGate Veridata supports both One-Way and Two-Way SSL connections.

This topic describes the following:

2.2.1 One-Way SSL Connection

In one-way SSL connection, the SSL client (Oracle GoldenGate Veridata Server) must trust the SSL server (Oracle GoldenGate Veridata Agent).

To enable SSL, you can use either self-signed certificates OR Certificate Authority (CA) signed certificates.

Using Self-Signed Certificates

1. Create self-signed certificates for all agents using keytool API. Keytool is a Java API that comes along with Java Installation. See [Generating Self-Signed Keystores and Truststores](#) for generating self-signed certificates.

2. Edit `agent.properties` for SSL configurations. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Agent](#).
3. Update Agent Wallet. See [Updating Oracle GoldenGate Veridata Agent Wallet](#).
4. Start Oracle GoldenGate Veridata Agent.
5. Edit Server Configuration file. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Server](#) (This step is optional. Use it if SSL connection should be established with all agents. For individual agents, SSL checkbox can be checked from UI under Connection Configuration)
6. Upload the TrustStore generated in Step 1 to the `VeridataWebTrustStore`. See [Updating Oracle GoldenGate Veridata Server Trust Store](#) .

Using CA Signed Certificates

1. Use certificates issued by the same CA for all agents.
2. Copy the certificates under `<Agent_Home>/config/certs` directory.
3. Edit `agent.properties` for SSL configurations. [Configuring SSL Settings for the Oracle GoldenGate Veridata Agent](#).
4. Update Agent Wallet. See [Updating Oracle GoldenGate Veridata Agent Wallet](#).
5. Start Oracle GoldenGate Veridata Agent.
6. Edit Server Configuration file. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Server](#) (This step is optional. Use it if SSL connection should be established with all agents. For individual agents, SSL checkbox can be checked from UI under Connection Configuration).
7. Trust the Agent root CA certificate in the server. See [Updating Oracle GoldenGate Veridata Server Trust Store](#) .

2.2.2 Two-Way SSL Connection

In two-way SSL, mutual trust is needed between Veridata Server and Agents for the SSL connection to be established. To enable SSL, you can use either self-signed certificates OR Certificate Authority (CA) signed certificates.

Using Self-Signed Certificates

1. Create self-signed certificates for all agents and Veridata Server. See [Generating Self-Signed Keystores and Truststores](#) for generating self-signed certificates.
2. Edit `agent.properties`. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Agent](#).
3. Update Veridata Agent Wallet. See [Updating Oracle GoldenGate Veridata Agent Wallet](#).
4. Start Agent(s).
5. Edit Server Configuration file. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Server](#).
6. Upload all agent certificates to the `VeridataWebTrustStore` and Server Identity certificate to `VeridataWebIdentityStore` of the Veridata Server. See [Updating Oracle GoldenGate Veridata Server Trust Store](#) .

Using CA Signed Certificates

1. Use certificates issued by the same CA for all agents.
2. Copy both AgentID and AgentTrust certificates under `<Agent_Home>/config/certs` directory.
3. Edit agent.properties. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Agent](#).
4. Update Veridata Agent Wallet. See [Updating Oracle GoldenGate Veridata Agent Wallet](#).
5. Start Veridata Agent(s).
6. Edit Server Configuration file. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Server](#).
7. Trust the root CA certificate in the server. See [Updating Oracle GoldenGate Veridata Server Trust Store](#).

2.2.3 Generating Self-Signed Keystores and Truststores

1. Build Agent Keystore:

```
keytool -genkey -alias agent.server.keys -keyalg RSA -keystore
vdtAgentIdentity.jks -storepass ks_password -keypass keypwd
```

2. Copy the `vdtAgentIdentity.jks` generated to `<Agent_Home>/config/certs` directory.
3. Export Agent Keystore to a certificate:

```
keytool -export -alias agent.server.keys -keystore vdtAgentIdentity.jks -
storepass ks_password -file agent.server.cer
```

4. Import the agent certificate to server truststore:

```
keytool -import -v -keystore serverTrust.jks -storepass ks_password -file
agent.server.cer
```

5. Copy `serverTrust.jks` to the box where Veridata Server is running.

The following points are for two-way SSL:

6. On a system where you have installed Oracle GoldenGate Veridata server, build the server keystore:

```
keytool -genkey -alias vdt.web.client.keys -keyalg RSA -keystore vdtServerID.jks -
storepass ks_password -keypass keypwd
```

7. Export the server certificate to a file:

```
keytool -export -alias vdt.web.client.keys -keystore vdtServerID.jks -storepass
ks_password -file vdt.web.client.cer
```

8. Import the server certificate to the agent truststore:

```
keytool -import -v -keystore vdtAgentTrust.jks -storepass ks_password -file
vdt.web.client.cer
```

Copy `vdtAgentTrust.jks` to all the Agent box under `<Agent_deployed_location>/config/certs`.

Following examples illustrate how to generate Keystore via Java Keytool:

Example 1: Create an Agent ID Keystore

```
keytool -genkey -alias vdt.agent.id -keyalg RSA -keystore
vdtAgentID.jks -storepass changeit -keypass changeit -validity 365
```

Copy

```
C:\java\Java8\jdk1.8.0_40\bin>keytool -genkey -alias vdt.agent.id -keyalg RSA -
keystore vdtAgentID.jks -storepass changeit -keypass c
hangeit -validity 365
```

What is your first and last name?

[Unknown]: COMPANY A

What is the name of your organizational unit?

[Unknown]: NA

What is the name of your organization?

[Unknown]: COMPANY A

What is the name of your City or Locality?

[Unknown]: USA

What is the name of your State or Province?

[Unknown]: USA

What is the two-letter country code for this unit?

[Unknown]: US

Is CN=COMPANY A, OU=NA, O=COMPANY A, L=USA, ST=USA, C=US correct?

```
keytool -export -alias vdt.agent.id -keystore vdtAgentID.jks -
storepass changeit -file vdtAgentID.cer
```

Copy

```
C:\java\Java8\jdk1.8.0_40\bin>keytool -export -alias vdt.agent.id -keystore
vdtAgentID.jks -storepass changeit
-file vdtAgentID.cer
```

The certificate is stored in the vdtAgentID.cer file.

Example 2 Create a Server ID Keystore

```
keytool -genkey -alias vdt.server.id -keyalg RSA -keystore
vdtServerID.jks -storepass changeit -keypass changeit -validity 365
```

Copy

```
C:\java\Java8\jdk1.8.0_40\bin>keytool -genkey -alias vdt.server.id -keyalg RSA -
keystore vdtServerID.jks -storepass changeit -keypass changeit -validity 365
```

What is your first and last name?

[Unknown]: ORACLE GOLDENGATE VERIDATA SERVER

What is the name of your organizational unit?

[Unknown]: NA

What is the name of your organization?

[Unknown]: COMPANY A

What is the name of your City or Locality?

[Unknown]: USA

What is the name of your State or Province?

[Unknown]: USA

What is the two-letter country code for this unit?

[Unknown]: US

Is CN=COMPANY A, OU=NA, O=COMPANY A, L=USA, ST=USA, C=US correct?

[no]: yes


```
keytool -export -alias vdt.server.id -keystore vdtServerID.jks
-storepass changeit -file vdtServerID.cer
```

Copy

```
C:\java\Java8\jdk1.8.0_40\bin>keytool -export -alias
vdt.server.id -keystore vdtServerID.jks
-storepass changeit -file vdtServerID.cer
```

The certificate is stored in the vdtServerID.cer file.

Create a TrustStore for Oracle GoldenGate Veridata Agent and Oracle GoldenGate Veridata Server

```
keytool -import -v
-keystore vdtAgentTrust.jks -storepass changeit -file vdtServerID.cer -
alias
vdt.server.id
```

Copy

```
C:\java\Java8\jdk1.8.0_40\bin>keytool -import -v -keystore vdtAgentTrust.jks -
storepass changeit -file vdtServerID.cer -alias vdt.ser
ver.id
Owner: CN=ORACLE GOLDENGATE VERIDATA SERVER, OU=NA, O=COMPANY A, L=USA, ST=USA, C=US
Issuer: CN=ORACLE GOLDENGATE VERIDATA SERVER, OU=NA, O=COMPANY A, L=USA, ST=USA, C=US
Serial number: 2aded02f
Valid from: Thu May 14 12:18:09 IST 2015 until: Fri May 13 12:18:09 IST 2016
Certificate fingerprints:
    MD5: 4E:7D:89:F7:C8:E8:64:37:E5:0C:D3:03:8F:3E:94:0A
    SHA1: 1B:00:9D:44:BD:73:6E:71:9D:44:56:4A:29:4E:F5:D7:1C:49:57:F3
    SHA256:
25:CB:77:3F:BC:5F:88:4B:09:D2:2D:C1:F8:E6:BA:70:DB:2B:55:53:48:7D:BA:F1:A3:01:18:AB:AA:
D1:56:6A
    Signature algorithm name: SHA256withRSA
    Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: EF C3 25 BB 83 4E 2D 0D 15 3D EF 50 F7 F2 D0 A6 ..%..N-..=.P....
0010: 94 5F 87 F2 ..
]
]
Trust this certificate? [no]: yes
Certificate was added to keystore
[Storing vdtAgentTrust.jks]
```

```
keytool -import -v -keystore vdtServerTrust.jks -storepass changeit
-file vdtAgentID.cer -alias vdt.agent.id
```

Copy

```
C:\java\Java8\jdk1.8.0_40\bin>keytool -import -v -keystore vdtServerTrust.jks -
storepass changeit -file vdtAgentID.cer -alias vdt.age
nt.id
Owner: CN=COMPANY A, OU=NA, O=COMPANY A, L=USA, ST=USA, C=US
Issuer: CN=COMPANY A, OU=NA, O=COMPANY A, L=USA, ST=USA, C=US
Serial number: 6b590df2
Valid from: Thu May 14 12:08:00 IST 2015 until: Fri May 13 12:08:00 IST 2016
Certificate fingerprints:
    MD5: 3E:75:A3:96:40:60:10:96:DD:10:7B:4D:E4:3F:4C:04
```

```

SHA1: D1:CC:EB:67:A1:C6:CD:CA:62:27:EA:F8:82:BF:AB:E4:E7:2B:45:6D
SHA256:
E7:20:CF:D4:48:E2:AE:1E:1C:C7:06:1A:B3:0A:17:1F:8F:02:88:B7:A6:A0:5D:F7:12:BC:26:
68:5B:C3:C9:C8
Signature algorithm name: SHA256withRSA
Version: 3

```

Extensions:

```

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: C0 D5 02 D9 24 6F 58 F6   63 D7 34 D3 9D C4 9E 33   ....$oX.c.4....3
0010: FC 16 4E 5F                               ..N_
]
]

```

```

Trust this certificate? [no]: yes
Certificate was added to keystore
[Storing vdtServerTrust.jks]

```

Renew the new Certificate from an Existing Certificate

To renew a new Truststore certificate, you need to first delete the existing truststore and then create a new TrustStore for Agent and Server

To delete the existing truststore from Oracle WebLogic by using the WLST script and creating a new Truststore:

1. Execute `<Oracle_Home>/oracle_common/bin/wlst` and execute the following commands to delete the existing certificate:

```

connect(url='t3://<host>:<admin port>')
svc = getOpssService(name='KeyStoreService')
svc.listKeyStores(appStripe='')
svc.deleteKeyStore(appStripe='stripe', name='keystore',
password='<password>')

```

2. After you have deleted it, create a new certificate by following the example **Create a TrustStore for Oracle GoldenGate Veridata Agent and Oracle GoldenGate Veridata Server**.

2.2.4 Configuring SSL Settings for the Oracle GoldenGate Veridata Agent

By default, SSL is disabled for the agent. To configure SSL, edit the following keys in the `agent.properties` file for your agent.

Table 2-1 SSL Settings in the `veridata.cfg` file

Parameter	Description	Default Value
<code>server.useSsl</code>	Enables or disables SSL communication between the agent and the server. Possible values are: true: Enables SSL communication false: Disables SSL communication	false

Table 2-1 (Cont.) SSL Settings in the veridata.cfg file

Parameter	Description	Default Value
server.use2WaySsl	Specifies whether the SSL communication value is one way or two way. Options are: <ul style="list-style-type: none"> true: Uses two-way SSL communication false: Uses one-way SSL communication 	false
server.identitystore.type	Specifies the type of keystore used for the SSL configuration.	JKS
server.identitystore.path	Specifies the path for the server identity keystore.	./config/certs/vdtAgentIdentity.jks (Self Signed)
server.truststore.path	Specifies the path for the server truststore.	./config/certs/serverTrust.jks (Self Signed)
server.identitystore.keyfactory.alg.name	Algorithm name of the keyfactory used for the SSL server identity store.	SunX509
server.truststore.keyfactory.alg.name	Algorithm name of the keyfactory used for SSL server trust store.	SunX509
server.ssl.algorithm.name	SSL algorithm name. Note: The value of this parameter must be same for the agent and the server.	TLS

2.2.5 Updating Oracle GoldenGate Veridata Agent Wallet

Ensure `agent.properties` is updated as explained in [Configuring SSL Settings for the Oracle GoldenGate Veridata Agent](#) before running this script. Run `configure_agent_ssl` script that is available in the agent home as follows:

```
AGENT_HOME\configure_agent_ssl.sh AgentID
```

- AgentID is the name of the agent properties file, without the `.properties` extension.

The following questions are prompted:

```
Enter Agent Identity Store unlock password :<Specify the storepassword >
Enter Agent Identity Store entry password: <Specify the keypassword>
Enter Agent Trust Store unlock password: <Specify the storepassword>
```

2.2.6 Configuring SSL Settings for the Oracle GoldenGate Veridata Server

To enable SSL communication for all server-agent connections, you must set the SSL parameters in the `veridata.cfg` file located in the `DOMAIN_HOME/config/veridata` directory of your installation. To establish SSL communication only for certain connections, edit the connection properties in the Oracle GoldenGate Veridata web user interface.

Table 2-2 SSL Settings in the veridata.cfg file

Parameter	Description	Default Value
server.useSsl	Enables or disables SSL communication between the agent and the server. Possible values are: <ul style="list-style-type: none"> true: Enables SSL communication false: Disables SSL communication 	false
server.ssl.client.identitystore.keyfactory.algorithm.name	Algorithm name of the keyfactory used for the SSL server identity store.	SunX509
server.ssl.client.truststore.keyfactory.algorithm.name	Algorithm name of the keyfactory used for the SSL server trust store.	SunX509
server.ssl.algorithm.name	SSL algorithm name. Note: The value of this parameter must be the same for the agent and the server.	TLS

2.2.7 Configuring an SSL Connection between Oracle GoldenGate Veridata Server and NonStop Agent

To configure an SSL Connection between Oracle GoldenGate Veridata Server and NonStop Agent:

1. Create a new certificate file for the Nonstop agent and name it `SSLCERT` to replace the sample certificate file that exists in the Nonstop agent.
2. On Veridata Server, generate a jks file based on a certificate file of NonStop agent from step 1 by using the following command:

```
keytool -import -v -keystore vdtServerTrust.jks -storepass
<create_your_own> -file SSLCERT -alias <create_your_own>
```

3. Edit the Server Configuration file. See [Configuring SSL Settings for the Oracle GoldenGate Veridata Server](#).
4. Follow the instructions in topic [Updating Oracle GoldenGate Veridata Server Trust Store](#) (except for step 6, to import the jks file to the Veridata server).

2.2.8 Updating Oracle GoldenGate Veridata Server Trust Store

1. Copy the Trust Certificate (`serverTrust.jks` in-case of SelfSigned/rootCA for CA Signed) generated earlier to any folder on Server.
2. Execute WLST script present under `<OracleHome>/oracle_common/common/bin`. This allows you to login to WLST tool in offline mode.

3. Connect to Admin Server from WLST console. Ensure Weblogic is up and running for the connection to be successful: `connect (url='t3://<host>:<port>')`. You are prompted for Weblogic credentials.

4. Run the following in WLST console:

```
svc = getOpssService(name='KeyStoreService')
svc.listKeyStores(appStripe='*')
```

The following should be seen in the output:

```
VeridataSec/VeridataWebTrustStore
VeridataSec/VeridataWebIdentityStore
```

5. Run the following command to import the generated Truststore in above step to VeridataWebTrustStore:

```
svc.importKeyStore(appStripe='VeridataSec', name='VeridataWebTrustStore',
                  password='storepass', aliases='agent.server.keys',
                  keypasswords='keypass', type='JKS',
                  permission=true, filepath='<path to Truststore file copied in step
1>')
```

6. In case of 2-way SSL, import ServerID to VeridataWebIdentityStore.

```
svc.importKeyStore(appStripe='VeridataSec', name='VeridataWebIdentityStore',
                  password='storepass', aliases='vdt.web.client.keys',
                  keypasswords='keypass', type='JKS',
                  permission=true, filepath='<path to ServerIdentity file created in
section
2.2.3>')
```

7. Run `exit()` command to exit WLST tool

8. Restart Admin and Managed Server.

2.3 Securing the Oracle GoldenGate Veridata Files

This topic describes how to secure your business data and control access to the Oracle GoldenGate Veridata installation directories and user interface.

Controlling Access to the Installation Directories

Standard operating system permissions apply to the programs, files, and directories in the server, agent, and web user interface installation directories. You should adjust the permissions for these objects based on your business security rules.

Securing Files That Contain Business Data

The server creates data files that contain sensitive application data. By default, these files reside in `DOMAIN_HOME/veridata/reports`. The subdirectories contain files that may reflect business data.

The following types of files contain sensitive data:

- Comparison report (`rpt` subdirectories)
- Out-of-sync report (`oosxml` and `oos` subdirectories)

These files inherit the same file permissions as those of the user that runs the server installation program. Do not change the permissions; if you do, then Oracle GoldenGate Veridata may be unable to maintain them. Keep these files just as secure as you would keep

your business data. Users of the Oracle GoldenGate Veridata web user interface do not require access to these files because they see the same information through the client interface. You can encrypt the contents of all report files.

2.4 Securing Access to Oracle GoldenGate Veridata by Defining User Groups

You assign security groups to control user access to the software functions, some of which expose selected data values from the database.

Oracle GoldenGate Veridata	Type	Description
veridataAdministrator	Type-A	The administrator role is the highest level security role. This role can perform all functions that configure, execute, and monitor Oracle GoldenGate Veridata.
veridataPowerUser	Type-A	The power user role is the second highest role. This role can perform all functions that configure, execute, and monitor Oracle GoldenGate Veridata from the web user interface. It cannot perform any configuration functions for the server.
veridataReportViewer	Type-B	The report viewer role cannot perform functions that configure Oracle GoldenGate Veridata or execute jobs. This role can only view configuration and job information, and view comparison reports.
veridataDetailReportViewer	Type-B	The detail report viewer role cannot perform any functions that configure Oracle GoldenGate Veridata or execute jobs. This role can only view configuration and job information. It can also view comparison reports and out-of-sync report information through the web user interface or at the file level.
veridataRepairOperator	Additional	The repair operator role can use the Repair feature in Oracle GoldenGate Veridata.
veridataCommandLineUser	Additional	The command-line user role provides access to the Oracle GoldenGate Veridata command-line tools, <code>vericom</code> , and the Oracle GoldenGate Veridata import and export utilities.

These roles are categorized as follows:

- Type A and Type B: By default, Type A and Type B users are not given any privileges of the additional user roles. Assign additional roles to users of these types.
- Additional: WebLogic Administrators can assign these Additional roles to Type A users to perform the required Oracle GoldenGate Veridata functions.

Security is controlled through the Oracle WebLogic Server Administration Console, where a user with the administrator role can:

- Create a user and assign it a security role.
- Create user groups and assign them security roles. Users can be added to these groups without being given a security role. A user inherits the role of its group.

- Create a user and assign it a security role, and then add that user to a group. The user inherits the role of its group and keeps its individual role.

Setting Up Active Directory Groups on Oracle GoldenGate Veridata

To set up Active Directory groups on Oracle GoldenGate Veridata:

1. In your browser, connect to the Oracle WebLogic Server Administration Console by entering the following address:

`http://weblogic_admin_server_hostname:admin_server_port/console`

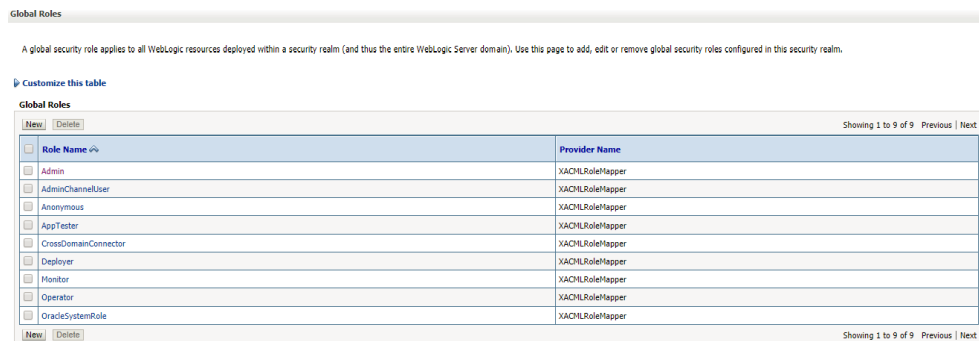
weblogic_admin_server_hostname is the name or IP address of the system where the server and web components are hosted, and *admin_server_port* is the port number assigned to the server (default is 7001).
2. Log on as an Oracle GoldenGate Veridata administrator user. A default administrator user was created when the Oracle GoldenGate Veridata domain was set up.
3. In the Oracle WebLogic Server Administration Console, navigate to **Security Realms**, then select **myrealm**, and then click **Roles and Policies**.
4. Select the **Realm Roles** subtab.
5. Expand **Global Roles** and then click on **Roles** link from the table.
6. Click **New** to create the following Roles:
 - a. ExtAdministrator
 - b. ExtPowerUser
 - c. ExtDetailReportViewer
 - d. ExtReportViewer
 - e. ExtRepairOperator

Oracle GoldenGate Veridata	Type	Description
ExtAdministrator	Type-A	The ExtAdministrator role is the highest-level security role in Oracle GoldenGate Veridata. This role can perform all of the functions that configure, execute, and monitor Oracle GoldenGate Veridata.
ExtPowerUser	Type-A	The ExtPowerUser role is the second-highest role in Oracle GoldenGate Veridata. This role can perform all of the functions that configure, execute, and monitor Oracle GoldenGate Veridata from the Oracle GoldenGate Veridata Web User Interface, but this role cannot perform any configuration functions for the Oracle GoldenGate Veridata Server.

Oracle GoldenGate Veridata	Type	Description
ExtReportViewer	Type-B	The ExtReportViewer role cannot perform functions that configure Oracle GoldenGate Veridata or execute jobs. This role can only view configuration and job information, and view comparison reports.
ExtDetailReportViewer	Type-B	The ExtDetailReportViewer role cannot perform any functions that configure Oracle GoldenGate Veridata or execute jobs. This role can only view configuration and job information, and view comparison reports and out-of-sync report information through the Oracle Oracle GoldenGate Veridata web user interface or at the file level.
ExtRepairOperator	Additional	The ExtRepairOperator role can use the Repair feature in Oracle GoldenGate Veridata.

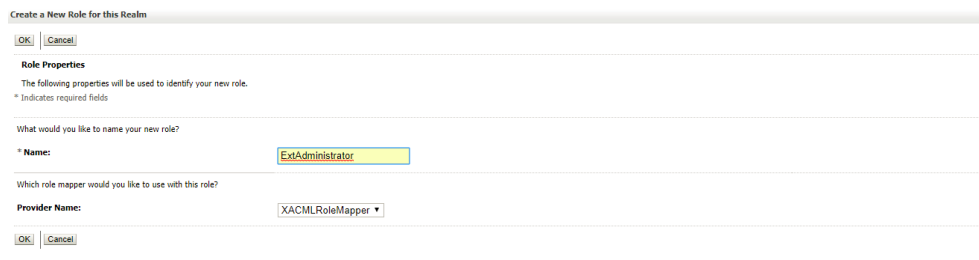
You can create a new role as shown in the following screen shot:

Figure 2-1 Global Roles



For example, consider **ExtAdministrator** role.

Figure 2-2 Example: ExtAdministrator role



7. Enter the **Role Name** and click **OK**.
8. Similarly, create all 5 Roles.
9. After all the Roles are created, navigate to **Global Roles Table** and select the role for which Active Directory group should be mapped and then click **Edit Role**, and then click **Next**.

Figure 2-3 Active Directory Roles

Name	Resource Type	Role Policy
Coherence Clusters		
Deployments		
Domain		
Global Roles		
Roles		
Admin	Global Role	View Role Conditions
AdminChannelUser	Global Role	View Role Conditions
Anonymous	Global Role	View Role Conditions
AppTester	Global Role	View Role Conditions
CrossDomainConnector	Global Role	View Role Conditions
Deployer	Global Role	View Role Conditions
ExtAdministrator	Global Role	View Role Conditions

10. Click **Add Conditions** button and then select **Group** from **Predicate List** and click **Next**.

Figure 2-4 Editing a Role

11. Enter the Active Directory group in **Group Argument Name**, click **Add**, click **Finish**, and then click **Save** to save the changes. For example, enter the **Group Argument Name** as **Finance**. After this group gets added, all users part of the group will have Oracle GoldenGate Veridata access.

Figure 2-5 Edit Arguments

2.5 Creating and Editing Users

You can create and edit users in the WebLogic Console.

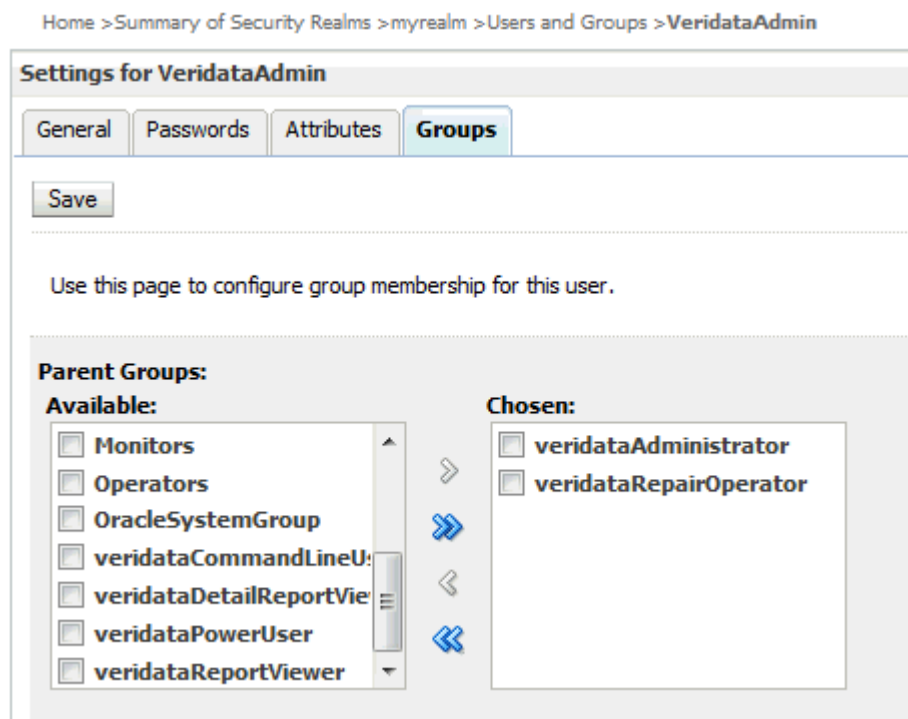
To create and edit users:

1. In your browser, connect to the Oracle WebLogic Server Administration Console by entering the following address:

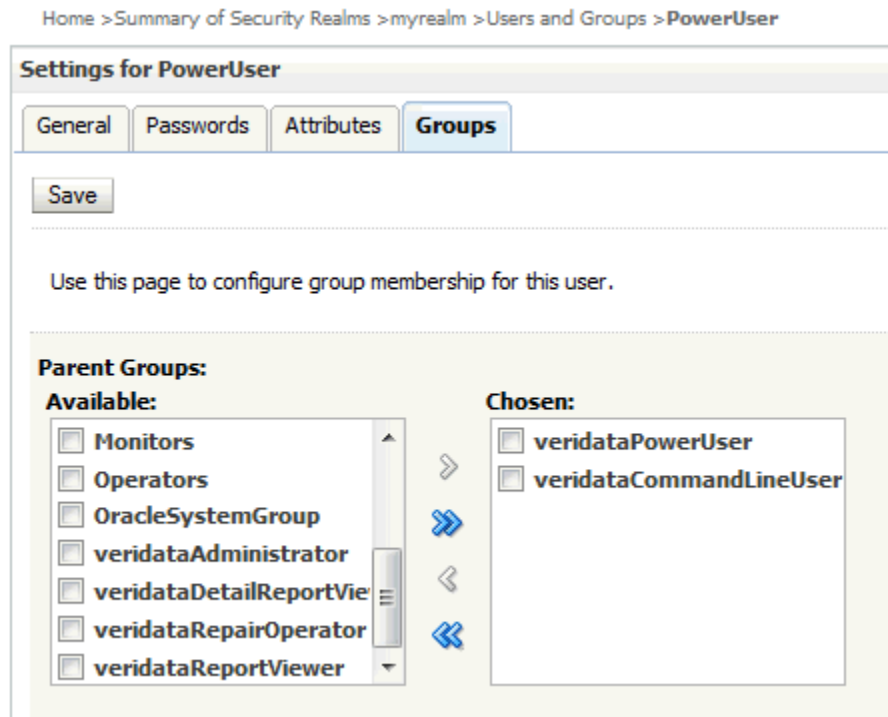
```
http://weblogic_admin_server_hostname:admin_server_port/console
```


weblogic_admin_server_hostname is the name or IP address of the system where the server and web components are hosted, and *admin_server_port* is the port number assigned to the server (default is 7001).
2. In the left pane of the Oracle WebLogic Server Administration Console, select **Security Realms**.
3. On the **Summary of Security Realms** page select the name of the security realm.
4. On the **Settings for Veridata Security** realm page, select **Users and Groups**, and then select **Users**. The User table displays the names of all users defined in the Authentication provider.
5. Select an existing user and edit the settings or click **New** to create a user and enter the user's properties.
6. To assign a role to the user, click **Groups** on the Settings for *user_name* page. Select appropriate roles for the user.

For example, an administrator, **VeridataAdmin**, can be given the privileges shown in the figure.



For example, a Veridata power user, **PowerUser**, can be given the privileges shown in the next figure.



7. Click **Save** .

2.6 Changing Database Schema Passwords

You can change database schema passwords when a schema password expires, an account is locked, or a password change is necessary. This topic applies to all database schemas that are prefixed with 'OGG', such as `OGG_IAU`, `OGG_IAU_APPEND`, or `OGG_IAU_VIEWER`.

To change a database schema password:

1. Shutdown the Administration Server and Oracle GoldenGate Veridata Server.

```
DOMAIN_HOME/veridata/veridata/bin/veridataServer.sh stop
```

2. Run the following sql queries to change the password and unlock the OPSS schema:

```
alter user
{SCHEMA_PREFIX}_IAU identified by {NEW_PASSWORD};
alter user {SCHEMA_PREFIX}
_IUA_APPEND identified by
{NEW_PASSWORD};
alter user {SCHEMA_PREFIX}_IAU_VIEWER identified by {NEW_PASSWORD}

;
alter user
{SCHEMA_PREFIX}_STB identified by {NEW_PASSWORD};
alter user {SCHEMA_PREFIX}
_VERIDATA identified by
{NEW_PASSWORD};
```

```
alter user {SCHEMA_PREFIX}_WLS identified by {NEW_PASSWORD}
;
alter user
{SCHEMA_PREFIX}_WLS_RUNTIME identified by {NEW_PASSWORD};

alter user {SCHEMA_PREFIX}

_OPSS identified by
{NEW_PASSWORD};
alter user {SCHEMA_PREFIX}_OPSS account unlock;
```

3. Navigate to the following directory:

```
{ORACLE_HOME}/oracle_common/common/bin . Run ./wlst.sh
```

4. Execute the following:

```
modifyBootStrapCredential (jpsConfigFile='{DOMAIN_HOME}/config/fmwconfig/jps-
co
nfig.xml',username='{SCHEMA_PREFIX}_OPSS',password='{NEW_PASSWORD}
')
```

username= is your database schema name and password= is the new password.

5. Execute: `exit()`.

6. Navigate to run:

```
{SCHEMA_PREFIX}

_STB details in "Database Configuration Type"
```

7. In the **Component Datasources page, select all the components and enter the new password if the password is same for all schemas . If the password is different for each schema, then select each schema and enter the password. Ensure to check, change password, and then uncheck for each component.**

8. Execute the `JDBC Test` and make sure it is successful.

9. Complete the configuration.

10. Start the Administration Server.

11. Start your Oracle GoldenGate Veridata Server.

```
DOMAIN_HOME/veridata/veridata/bin/veridataServer.sh start
```

12. Login to verify.

2.7 Encrypting Report Files

You can encrypt the comparison report files (`.rpt`, `.oos`, `.oosxml`) in Oracle GoldenGate Veridata. The following sections explain report encryption:

- [Enabling Report Encryption](#)
- [Displaying Encrypted Files with the `reportutil` Utility](#)

2.7.1 Enabling Report Encryption

The encryption is controlled by the following parameters in the `veridata.cfg` configuration file:

- `server.encryption`
- `server.encryption.bits`

To enable encryption, set `server.encryption` to `true`.

Encryption of report files uses Advanced Encryption Standard (AES) encryption, and the default encryption strength is 128 bits. You can increase the encryption strength to 192 or 256 bits by editing the value of the `server.encryption.bits` parameter in `veridata.cfg`. However, for encryption strength greater than 128, you must use a JRE installed with the Unlimited Strength Cryptography Extension.

For more information about these parameters, see "[Parameters for Report File Encryption](#)".

Encrypted report files have the following extensions in the file names:

- `.xrpt` : Encrypted comparison or repair report file
- `.xoos`: Encrypted binary out-of-sync file
- `.xoosxml`: Encrypted out-of-sync XML file
- `.xNNN`: Encrypted out-of-sync JSON chunk file (`NNN` is a decimal number)

2.7.2 Displaying Encrypted Files with the `reportutil` Utility

When report encryption is enabled, all report files are encrypted by using an encryption key, which is initially a large random value. If necessary, then you can change the encryption key.

Before you can read encrypted files, you must decrypt them. The Oracle GoldenGate Veridata web user interface automatically decrypts files before displaying them. Use the `reportutil.sh/.bat` utility to display the encrypted contents. This utility is located in the `VERIDATA_DOMAIN_HOME\veridata\bin` directory.

To display encrypted files, run the `reportutil.sh/.bat` utility located in the `VERIDATA_DOMAIN_HOME\veridata\bin` directory.

```
reportutil [-wlport port] -wuser weblogic_user { options }
```

`wlport` is the server port number (the default port is 8830), and `wuser` is the server user name.

The valid options are:

- `-version, -v`: Displays the current version
- `-help`: Displays the help message
- `-r`: Rolls report encryption
- `-f filename [-d directory]`: Decrypts and prints the report file to the specified file if a directory is specified by the `-d` option. Otherwise, the command prints the decrypted file to the standard output.

The user running the `reportutil` utility must be in the appropriate user group to perform the operations:

- `-r, -f`: Allowed only if the user is a member of the `veridataCommandLineUser` group.
- `-r`: Allowed if the user is a member of the `veridataAdministrator` group.
- `-f`: Allowed if the user is a member of the `veridataAdministrator` group, or a member of `veridataPowerUser` group, the `veridataPowerUser` group, or the `veridataDetailReportViewer` group.

For more information about the user roles, see [Securing Access to Oracle GoldenGate Veridata by Defining User Groups](#).

3

Managing C-Agent, Manager, and Java Components

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

This chapter includes the following sections:

3.1 Starting and Stopping the C-Agent and the Oracle GoldenGate Veridata Manager

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 on all platforms

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

```
START MANAGER
```

```
STOP MANAGER
```

3.2 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 agent component:

1. Switch to its installation directory.

2. Start or stop the agent.

UNIX or LINUX

```
agent.sh {start | run} OR agent.sh stop
```

Windows

```
agent.bat {start | run} OR agent.bat 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.

3.3 Reloading Logging Information

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

To reload logging information, start the agent and run the `reloadLog` commands shown in the following table:

UNIX and Linux	Windows
<code>agent.sh reloadLog</code>	<code>agent.bat reloadLog</code>

3.4 Controlling Logging Levels for Oracle GoldenGate Veridata Agent

To control the agent logging levels so that they log more information for debugging purposes, update the `AGENT_ORACLE_HOME/config/odl.xml` configuration file. Relevant comments are provided in the `odl.xml` file.

3.5 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 8830). Use `localhost` as the host name if you are connecting on the system that is local to the server installation.

Examples:

```
http://localhost:8830/veridata  
http://sysa:8830/veridata
```

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.

4

Managing Identities and Comparing Data

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

This chapter includes the following sections:

4.1 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 compare pair of a job

 **Note:**

You cannot run a group individually.

- Set tracing (only under guidance of an Oracle Support analyst)

For specific compare pairs, you can:

- Review previous out-of-sync results
- Generate out-of-sync XML from the previous run
- 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.

For more information, see [Using the Oracle GoldenGate Veridata Command-line Utilities-Vericom, Veridata GoldenGate Parameter Processing, and Import/Export](#).

4.2 Running the Vericom Tool

Anyone who has operating system permissions can run the `vericom` tool.

1. On the system where Oracle GoldenGate Veridata is installed, run the operating system's command shell.
2. Navigate to the `VERIDATA_DOMAIN_HOME/veridata/bin` directory.
3. Run the `vericom` tool with the following parameters:

Basic Parameters

Enter one of the following required options:

- **Run the Vericom tool:** `/vericom.sh -wuser <wusername> -wport <wport>.`
For example: `/vericom.sh -wuser vericom -wport 8830`
- **Run the command to identify the Oracle GoldenGate Veridata version:** `/vericom.sh -wuser <wusername> -wport <wport> -version.` For example: `/vericom.sh -wuser vericom -wport 8830`
- **Run a job:** `/vericom.sh -wuser <wuser> -wport <wport> -job <job name> -g <group name> -c <compare pair>`
For example: `./vericom.sh -wuser vericom -wport 8830 -job Vericom -g Vericom -c DP=DP`

Sample output:

```

OGGV-20032: Run ID: (7979,0,0)
OGGV-20033: Number of Rows Compared: 5
OGGV-20034: Number of Rows In Sync: 1
OGGV-20035: Number of Rows With Errors: 0
OGGV-20036: Number of Rows Out Of Sync: 4
OGGV-20037: Compare Pair OOSXML Directory: /scratch/username/
Oracle_Home/user_projects/domains/base_domain1/veridata/reports/
oosxml/Vericom/00007979/Vericom
OGGV-20038: Compare Pair OOSXML Filename:
OGGV-20039: Job Completion Status: With Out-Of-Sync
bash-4.4$

```

- **Purge a job:** For example: `/vericom.sh -wuser vericom -wport 8830 -purgeJobReports job1,job2,job3`
The `purgeJobReports all` parameter deletes all the latest jobs and reports.
- **Run the Help Command:** `/vericom.sh -wuser <wusername> -wport <wport> -version -help`
For example: `/vericom.sh -wuser vericom -wport 8830 -help`

Sample output:

```

Oracle GoldenGate Veridata Command Line Interface - Version 12.2.1.4.0
190913 release build 19
vericom [-wport <port> ] -wuser <user Name> { -job | -j } <job>
[run_flags]
                | -help | { -version | -v}
                | -helprun

```

Where:

```

[-wport <port> ] Weblogic port number
-wuser <user Name> Weblogic User Name to connect
{ -job | -j } <job>
    Run job <job>
-help
    Print usage.
-version, -v
    Print version.
-helprun
    Print usage of run_flags
run_flags
    Flags for running a job or compare pair

```

```

-addCredentialStore
    Create new Credential Store at location defined in veridata.cfg file
-deleteCredentialStore
    Delete Credential Store
-createAlias
    Create alias for user provided in wluser argument
-updateAlias
    Update username and password for alias with user provided in wluser argument
-deleteAlias
    Delete alias
-displayAlias
    Display alias stored in the credential store
-wlUserAlias
    Use alias inplace of wluser

```

- **Run the Help Run Command:** `/vericom.sh -wluser <wlusername> -wlport <wlport> -version -helprun`
For example: `/vericom.sh -wluser vericom -wlport 8830 -helprun`

Sample output:

```

Oracle GoldenGate Veridata Command Line Interface - Version 12.2.1.4.0 190913
release build 19
vericom [-wlport <port> ] -wluser <user Name> { -job | -j } <job>
[run_flags]
                | -help | { -version | -v}
                | -helprun

```

Where:

```

[-wlport <port> ] Weblogic port number
-wluser <user Name> Weblogic User Name to connect
{ -job | -j } <job>
    Run job <job>
-help
    Print usage.
-version, -v
    Print version.
-helprun
    Print usage of run_flags
run_flags
    Flags for running a job or compare pair
-addCredentialStore
    Create new Credential Store at location defined in veridata.cfg file
-deleteCredentialStore
    Delete Credential Store
-createAlias
    Create alias for user provided in wluser argument
-updateAlias
    Update username and password for alias with user provided in wluser argument
-deleteAlias
    Delete alias
-displayAlias
    Display alias stored in credential store
-wlUserAlias
    Use alias inplace of wluser

```

Run Flags Usage:

```

[ -g group_name -c compare_pair_name ]
[ -rP profile_name ]
[ -rR | -rO ]
[ -rN number_of_threads ]
[ -rD number_of_seconds ]

```

```
[ -rC | +rC ]
[ -rOb | -rOx | -rO2 | -rO0 ]
[ -rOs number_of_rows ]
[ -rd0 | -rdN run_id ]
[ -pS partition_name
  | -pSq SQL_predicate
  | -pSA1 ascii_start_key
  | -pSA2 ascii_ending_key
  | -pSH1 hex_start_key
  | -pSH2 hex_ending_key ]
[ -pT partition_name
  | -pTq SQL_predicate
  | -pTA1 ascii_start_key
  | -pTA2 ascii_ending_key
  | -pTH1 hex_start_key
  | -pTH2 hex_ending_key ]
[ -pq SQL_predicate ]
[-nw ]
[ -w | -wp interval ]
[ -repair ]
[ -norepair ]
```

Where:

```
-g group_name
    Name of the group. Requires -c.
-c compare_pair_name
    Name of the compare pair. Requires -g.
-rP profile_name
    Override the default profile.
-rR    Review Previous OOS Results.
-rO    Generate OOS XML from Previous Run.
-rd0   Do not use delta base from previous run
        for this job. Requires -c.
-rdN run_id
    Use delta base from run run_id.
        Requires -c.
-rN number_of_threads
    Override the number of concurrent threads.
-rD number_of_seconds
    Override the COOS delay for this job.
-rC    Do not run COOS this job.
+rC    Do run COOS this job.
-rOb   Generate binary OOS for this job.
-rOx   Generate XML OOS for this job.
-rO2   Generate both binary and XML OOS for this job.
-rO0   Do not generate OOS for this job.
-rOs number_of_rows
    Number of rows in each OOS XML chunk for this job.
-pS    Use named partition on the source.
-pSq   Use SQL predicate on the source.
-pSA1  Use arg as the starting ascii key the source.
-pSA2  Use arg as the ending ascii key the source.
-pSH1  Use arg as the starting hexadecimal key the source.
-pSH2  Use arg as the ending hexadecimal key the source.
-pT    Use named partition on the target.
-pTq   Use SQL predicate on the target.
-pTA1  Use arg as the starting ascii key the target.
-pTA2  Use arg as the ending ascii key the target.
-pTH1  Use arg as the starting hexadecimal key the target.
-pTH2  Use arg as the ending hexadecimal key the target.
-pq    Use SQL predicate on the source and target.
-nw    Do not wait for job to complete; job is submitted in background.
```

```

-wp      Wait for job to complete, poll at specified interval.
-repair  Launch a repair job automatically after compare job finishes. ]
-norepair Do not launch a repair job automatically, after compare job finishes. ]
-repairsql Supports generation of repair sql after the Job comparison is
completed. When the compare is complete, a repair sql zip file gets generated if
out-of-sync data exists. This flag is valid only for Oracle and SQL Server target
databases. ]

```

See [Securing Access to Oracle GoldenGate Veridata by Defining User Groups](#).

The `-version`, `-v`, `-help`, or `-helprun` options, they take precedence over any other option specified.

Table 4-1 Explaining the Vericom Basic Parameters

Argument	Description
<code>-wuser</code>	Specifies the server user name that authenticates and connects to the server.
<code>-ssl</code>	Used to connect to Oracle GoldenGate Veridata Server through the SSL.
<code>-wport</code>	Specifies the server port number.
<code>-help</code>	Displays the <code>vericom</code> syntax components and their descriptions.
<code>-helprun</code>	Displays run-related syntax components and their descriptions.
<code>{-version -v}</code>	Displays the version of the Oracle GoldenGate Veridata command-line interface that is being used.
<code>{-job -j} job</code>	Specifies the job to be run. For <i>job</i> , specify the name that was assigned when the job was created in Oracle GoldenGate Veridata Web.
<code>-g group -c compare_pair</code>	Specifies a group and compare pair. For <i>group</i> and <i>compare_pair</i> , specify the names that were assigned when these objects were created in Oracle GoldenGate Veridata Web. <ul style="list-style-type: none"> If <code>-g</code> and <code>-c</code> are used, <code>-j</code> must also be used.

Table 4-2 Explaining the Vericom Advanced Parameters

Argument	Description
<code>-repair -norepair</code>	Specifies whether to repair after a comparison is completed and confirms that out-of-sync data exists
<code>-repairsql</code>	Supports generation of repair sql after the Job comparison is completed. When the compare is complete, a repair sql zip file gets generated if out-of-sync data exists. By default, the file is generated in the <code>/tmp</code> directory and the location gets printed on console when Vericom is run. If you want to change the location, then you can modify it here: Profile -> Repair -> Repair SQL Path. This flag is valid only for Oracle and SQL Server target databases.

Table 4-2 (Cont.) Explaining the Vericom Advanced Parameters

Argument	Description
<code>-rP profile</code>	<p>Overrides the defined for a job. For profile that is defined for a job. For <i>profile</i>, specify the name that was assigned when the profile was created in Oracle GoldenGate Veridata Web.</p> <p>If <code>-rP</code> is used, then <code>-j</code> must be used.</p>
<code>-rR</code>	<p>A run override option. Compares only those rows that were out-of-sync in the previous run, based on the information that is stored in the out-of-sync file. The results identify which rows were brought back into synchronization by replication or another method.</p> <ul style="list-style-type: none"> Don't use <code>-rR</code> and <code>-rO</code> in the same run.
<code>-rO</code>	<p>A run override option. Generates an OOSXML file that is based on the out-of-sync file from the previous run. It generates XML for every row that is in the file. You can use the XML to view the out-of-sync information in an XML editor or for other purposes.</p> <ul style="list-style-type: none"> You must use <code>-rO</code> with <code>-j</code>. Don't use <code>-rR</code> and <code>-rO</code> in the same run.
<code>-rN threads</code>	<p>Specifies the number of concurrent comparison threads to use. You can use as many threads as there are processors on the server system. This option overrides the default job profile and has no effect if a job is not run with <code>-j</code> or if just one comparison is run by using <code>-j</code> with <code>-g</code> and <code>-c</code>.</p>
<code>-rD seconds</code>	<p>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. This option overrides the default job profile and has no effect if you use the <code>-rR</code> option.</p>
<code>-rC</code> <code>+rC</code>	<p>Controls whether or not the confirmation step (confirm OOS) is performed in the job.</p> <ul style="list-style-type: none"> <code>-rC</code> skips the confirmation step. You can skip the confirmation step if activity on the source tables is stopped or if replication doesn't continuously update the target tables. <code>+rC</code> includes the confirmation step. <p>These options override the default job profile and are mutually exclusive. They have no effect unless you use <code>-j</code>.</p>
<code>-rOb</code> <code>-rOx</code> <code>-rO2</code> <code>-rO0</code>	<p>Controls the kind of file that is produced for the out-of-sync report.</p> <ul style="list-style-type: none"> <code>-rOb</code> generates binary format that is compatible with the Oracle GoldenGate Veridata Web browser. <code>-rOx</code> generates output in XML. <code>-rO2</code> generates both binary and XML output. <code>-rO0</code> suppresses out-of-sync output. <p>These options override the default job profile and are mutually exclusive. They have no effect if you use <code>-rR</code>.</p>

Table 4-2 (Cont.) Explaining the Vericom Advanced Parameters

Argument	Description
<code>-rOs records</code>	Limits the number of out-of-sync rows that are written to a chunk of the <code>00SXML</code> file. Writing the file in chunks prevents it from becoming too large for the system to manage and allows periodic archiving or purging. The current file is closed when the specified number of rows is written, and a new file is opened. This option overrides the default job profile and has no effect if you use <code>-rR</code> .
<code>-rTi</code>	Turns on tracing of Oracle GoldenGate Veridata Agent (agent) for the initial comparison step. Don't use it without the guidance of an Oracle support analyst.
<code>-rTc</code>	Turns on tracing of the agent for the confirmation step. Don't use it without the guidance of an Oracle support analyst.
<code>-rTs trace_number</code>	Turns on tracing for the server. <code>trace_number</code> is a bitmask of server execution trace flags. A higher level of trace flags results in more detailed trace data.

Table 4-2 (Cont.) Explaining the Vericom Advanced Parameters

Argument	Description
<p>-pS <i>source_partition_name</i> -pSq <i>source_sql_predicate</i> -pSA1 <i>source_ascii_start_key</i> -pSA2 <i>source_ascii_end_key</i> -pSH1 <i>source_hex_start_key</i> -pSH2 <i>source_hex_end_key</i></p>	<p>Runs the comparison using an existing source row partition or using an override partition that is defined by partition criteria. These options are mutually exclusive. They are valid only if comparing one compare pair (-j with -g and -c) and are ignored otherwise.</p> <p>-pS <i>source_partition_name</i> Specifies an existing source partition that is already defined and stored in the repository. The partition name is not validated and is passed directly to the server. An error is returned if the specified partition does not exist.</p> <p>-pSq <i>source_sql_predicate</i> Specifies a SQL predicate that defines a partition to override an existing source partition for a SQL table. The predicate is the conditional statement that follows the WHERE keyword, for example: LAST_NAME BETWEEN "A" AND "M". Do not include the WHERE keyword. It will be added automatically at runtime.</p> <p>If the predicate contains multiple words, it must be enclosed within quotes to make it a single command argument. The type of quote is dependent on the command shell or interpreter that is being used.</p> <p>If the predicate contains special characters (such as \$, *, < in sh/csh or %, < in Windows), then they must be properly escaped for that shell or interpreter.</p> <p>-pSA1 <i>source_ascii_start_key</i> Specifies an ASCII key as the starting key value of a partition that overrides an existing source partition for an Enscribe file.</p> <p>-pSA2 <i>source_ascii_end_key</i> Specifies an ASCII key as the ending key value of a partition that overrides an existing source partition for an Enscribe file.</p> <p>-pSH1 <i>source_hex_start_key</i> Specifies a hexadecimal key as the starting key value of a partition that overrides an existing source partition for an Enscribe file.</p> <p>-pSH2 <i>source_hex_end_key</i> Specifies a hexadecimal key as the ending key value of a partition that overrides an existing source partition for an Enscribe file.</p>
<p>-pT <i>target_partition_name</i> -pTq <i>target_sql_predicate</i> -pTA1 <i>target_ascii_start_key</i> -pTA2 <i>target_ascii_end_key</i> -pTH1 <i>target_hex_start_key</i> -pTH2 <i>target_hex_end_key</i></p>	<p>These options specify target partitions and have the same rules as the corresponding options that specify source partitions.</p>

Table 4-2 (Cont.) Explaining the Vericom Advanced Parameters

Argument	Description
<code>-pq sql_predicate</code>	Specifies a SQL predicate to be used for both the source and target SQL tables, as an override to existing partitions. This option has the same rules as <code>-pSq</code> <i>source_sql_predicate</i> and <code>-pTq</code> <i>target_sql_predicate</i> .
<code>-rd0</code> <code>-rdN run_ID</code>	Controls delta processing for a compare pair. <ul style="list-style-type: none"> <code>-rd0</code> disables delta processing for this run. All rows are compared. <code>-rdN run_ID</code> enables delta processing by using a previous job run as the basis for the delta. For <i>run_ID</i> use the number from the Run ID line at the beginning of the job comparison report. Vericom does not validate the run ID that is supplied. To use these options, you must specify a compare pair with: <ul style="list-style-type: none"> <code>-j</code> <code>-g</code> <code>-c</code>
<code>-wp seconds</code>	Waits for a job to complete. The client also polls of the status of job submitted to the server at the specified interval (in seconds). <ul style="list-style-type: none"> The argument accepts no data or a single data as input. If the input is provided, then it is used as an alias name. If no input is provided, then the user name provided in the <code>wuser</code> argument is taken as an alias. This alias is used in place of the user name and password. You do not have to provide the actual user name and password. <code>-wp</code> must be used with <code>-job</code> <code>-j</code> .
<code>-addCredentialStore</code>	This argument does not accept any input. The default location for credential store is <code><Domain_home>/veridata/dircred</code> . You can change the directory by specifying it in the <code>veridata.cfg</code> file under property <code>credential.store.location</code> . The default value of <code>credential.store.location</code> property is <code>veridata/dircred</code> which is relative to the domain location. You can create your own credential store location and change this property before running vericom. The credential store wallet is created with only read and write permission (<code>-rw-----</code>).
<code>-deleteCredentialStore</code>	This argument does not accept any input. Use this option to delete the credential store. The location for credential store is the value of property <code>credential.store.location</code> defined in <code>veridata.cfg</code> file. The credential store wallet and its contents will be permanently deleted.
<code>-createAlias</code>	The argument accepts no data or a single data as input. If the input is provided, then it is used as an alias name. If no input is provided, then the user name provided in the <code>wuser</code> argument is taken as an alias. This alias is used in place of the user name and password. You do not have to provide the actual user name and password.

Table 4-2 (Cont.) Explaining the Vericom Advanced Parameters

Argument	Description
-wuserAlias	This argument is used with an alias that is created by using <code>-createAlias</code> in place of user name. You aren't prompted for a password. You aren't required to use this option with <code>-wuser</code> argument. If the alias does not exist in the wallet then an error is returned.
-displayAlias	Use this argument to list all aliases in the wallet. Only the alias and user name are displayed.
-updateAlias	Use this argument is used to update your password.
-deleteAlias	Use this argument to delete the alias that you created in the wallet.

4.2.1 SSL Configuration for Vericom

The Vericom tool is now supported to communicate with Oracle GoldenGate Veridata server using SSL protocol.

To configure SSL for Vericom

1. Generate the keystore for the server and the client using the JDK keytool utility that is available at `JAVA_HOME/bin` as shown in the following sample script.

```
#####
# Creating Server and Client KeyStores.
#####
# NOTE: "keytool" is a utility that can be found inside the "$JAVA_HOME/bin"
# so make sure the PATH includes the JDK's bin directory.

mkdir Keystores
cd Keystores/

echo ""
echo ""
echo "Creating WLS Server side Keystore. (wls12c.keystore)"
echo "-----"
keytool -genkey -v -alias wlsalias -keyalg RSA -keysize 1024 -keystore
wls12c.keystore -validity 3650 -keypass <keypassword> -storepass
<storepassword>+magic -dname "CN=127.0.0.1, OU=MiddlewareMagic, O=Blog,
L=Bangalore, S=Karnataka, C=IN"

echo ""
echo ""
echo "Exporting public key (wls12c_server.cer) from the WLS ServerSide
keystore."
echo "-----"
keytool -export -keystore wls12c.keystore -alias wlsalias -file
wls12c_server.cer -keypass <keypassword> -storepass <storepassword>+magic

echo ""
echo ""
echo "Creating Client side Keystore/truststore. (clientTrustStore.keystore)"
echo "-----"
```

```
keytool -genkey -v -alias clientalias -keyalg RSA -keysize 1024 -keystore
clientTrustStore.keystore -validity 3650 -keypass <keypassword> -storepass
<storepassword> -dname "CN=127.0.0.1, OU=MiddlewareMagic, O=Blog, L=Bangalore,
S=Karnataka, C=IN"

echo ""
echo ""
echo "Importing the WLS Servers public key to the Client's truststore."
echo "-----"
keytool -import -v -trustcacerts -alias wlsalias -file wls12c_server.cer -keystore
clientTrustStore.keystore -keypass <keypassword> -storepass <storepassword>

echo "Certificates created Successfully !!!"
```

2. In the WebLogic Server console, go to **Environment -> Servers -> VERIDATA_server1 -> Keystores** tab, enter the following values, and click **Save**:

- **Keystores:** Custom Identity and Java Standard Trust
- **Custom Identity Keystore:** /home/oracle/Oracle/Middleware/Oracle_Home/user_projects/domains/base_domain/veridata/dircred/wls12c.keystore
- **Custom Identity Keystore Type:** JKS
- **Custom Identity Keystore Passphrase:** [password set during keytool commands]
- **Confirm Custom Identity Keystore Passphrase:** [password set during keytool commands]

3. Click **Environment -> Servers -> VERIDATA_server1 -> SSL** tab, enter the following values and click **Save**.

- **Private Key Alias:** Custom Identity and Java Standard Trust
- **Private Key Passphrase:** [password set during keytool commands]
- **Confirm Private Key Passphrase:** [password set during keytool commands]

4. Enter the absolute path location for the client keystore in the `veridata.cfg` configuration file as a key/value pair. For example: `cli.ssl.client.trustStorePath = /scratch/<userName>/java/jdk1.8.0_251/bin/Keystores/clientTrustStore.keystore`.

5. Navigate to the `VERIDATA_DOMAIN_HOME/veridata/bin` directory.

6. Run the `vericom` tool to store the client store password as shown in the following example:

```
./vericom.sh -wuser veridata -wlport 8830 -saveTrustStorePassword
<password>
```

7. Run the `vericom` in `ssl` port by passing the `-ssl` parameter as shown in the following example:

```
./vericom.sh -wuser veridata -wlport 8830 -j Vericom -ssl
```

4.3 Managing Identities in a Credential Store

The credential store is implemented as an Auto Login wallet within the Oracle Credential Store Framework.

This section shows you how to use a credential store to maintain encrypted database passwords and user IDs and associate them with an alias. It is the alias, not the actual user

ID or password, that is specified in a command or parameter file. No user input of an encryption key is required.

Credential Store contains the following topics:

- [Adding a Credential Store](#)
- [Deleting a Credential Store](#)

Alias contains the following topics:

4.3.1 Adding a Credential Store

The `-addCredentialStore` argument does not accept any input. The default location for credential store is `<Domain_home>/veridata/dircred`. You can change this by specifying a directory in the `veridata.cfg` file under the `credential.store.location` property. The default value of the `credential.store.location` property is `veridata/dircred` which is relative to the domain location. You can create your own credential store location and change this property before running `vericom`. The credential store wallet is created with only read and write permissions (`-rw-----`).

Example 4-1

```
./vericom.sh -wuser vericom -wport 8830 -addCredentialStore
```

4.3.2 Creating an Alias

The `-createAlias` argument accepts zero or one as input. If the input is provided, then it is used as an alias name. If it is not provided, then the user name provided in the `wuser` argument is used as an alias. This alias is used in place of the user name and password, and you do not have to provide an actual user name and password.

Example 4-2

```
./vericom.sh -wuser vericom -wport 8830 -createAlias
```

4.3.3 Deleting a Credential Store

The `-deleteCredentialStore` argument does not accept any input. This option deletes the credential store. The location for the credential store is `credential.store.location` in the `veridata.cfg` file. The credential store wallet and its contents are permanently deleted.

Example 4-3

```
./vericom.sh -wuser vericom -wport 8830 -deleteCredentialStore
```

4.3.4 Using the Alias

You must use the `-wuserAlias` argument with the alias that you created with `-createAlias`. With this option, you are not prompted for a password. You should not use the `-wuserAlias` argument with `-wuserAlias`. If the alias does not exist in the wallet, then an error is returned.

Example 4-4

```
./vericom.sh -wuserAlias vericom -wport 8830 -job Vericom
```

4.3.5 Display Alias

Use the `-displayAlias` argument to list all aliases and user names in the wallet. The password is not displayed.

Example 4-5

```
vericom.sh -wluser vericom -wlport 8830 -displayAlias
```

Sample Output

```
Alias : vericom
Username : vericom
```

4.3.6 Updating the Alias

Use the `-updateAlias` argument to update the password for the valid user.

Example 4-6

```
./vericom.sh -wluser vericom -wlport 8830 -updateAlias
```

Sample Output:

Successfully updated alias 'vericom' for user.

4.3.7 Deleting the Alias

The `-deleteAlias` argument used to delete the alias. Along with the alias, the corresponding user credentials are also removed if, no other alias is referring to the same user.

Example 4-7

```
./vericom.sh -wluser vericom -wlport 8830 -deleteAlias
```

4.4 Vericom Exit Statuses

The `vericom` command-line tool exits with one of the following statuses. The examples are for a UNIX or Linux system.

Vericom exits with one of the following statuses. This examples shown are for a UNIX or Linux system.

Table 4-3 Vericom Exit Status

Status	Description
0	The command executed successfully. If a job was run, then all rows are in-sync. If you specified <code>-nw</code> , then the exit status is 0 if the job started successfully.
1	Invalid <code>vericom</code> syntax was used. For example, the following are invalid: <code>vericom.sh -helptun</code> (A typographical error occurred.) <code>vericom.sh -j -g group1</code> (The name of the job is missing.)

Table 4-3 (Cont.) Vericom Exit Status

Status	Description
3	<p>Provides more granularity for input errors that involve comparison flags. For example, the following mistakes cause this error:</p> <pre>vericom.sh -j job1 -c address=address</pre> <p>In the preceding example, the <code>-g group</code> input is missing. It is required with <code>-j</code> if <code>-c</code> is used.</p> <pre>vericom.sh -j job1 -g group1 -rd0</pre> <p>In the preceding example, the <code>-rd0</code> flag requires <code>-c</code> because delta processing applies at the compare pair level.</p>
4	The job contains rows that are out of sync.
5	There was a communication error with the server.
6	The job has failed.
7	The job was cancelled.

4.5 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.

See [Viewing Comparison Results](#).

The following examples use the `TestJob` job:

Example 1

This example shows a run on a Windows system without specifying `-w`. The process exits with status 0, and finished job statistics are not displayed.

```
VERIDATA_DOMAIN_HOME\veridata\bin\vericom.bat -wuser veridata -wlpport 8830 -j
TestJob
Connecting to: localhost:9177
Run ID: (2256, 0, 0)
C:\veridata\server\bin> if errorlevel 0 echo EXITED 0 STATUS
EXITED 0 STATUS
```

Example 2

This example shows a run of the `TestJob` with `-w` specified. The process exits with status 4 because one of the compare pairs had a validation error. Finished job statistics are displayed.

```
VERIDATA_DOMAIN_HOME\veridata\bin\vericom.bat -wuser veridata -wlpport 8830 -j
TestJob -w
Connecting to: localhost:9177
```

```
Run ID: (2257, 0, 0)
Job Start Time: 2008-03-21 22:48:05
Job Stop Time: 2008-03-21 22:48:20
Job Report Filename: C:\testjunit\rpt\TestJob\00002257\TestJob.rpt
Number of Compare Pairs: 3
Number of Compare Pairs With Errors: 1
Number of Compare Pairs With OOS: 1
Number of Compare Pairs With No OOS: 1
Number of Compare Pairs Cancelled: 0
Job Completion Status: WITH ERRORS
C:\veridata\server\bin> if errorlevel 4 echo EXITED 4 STATUS
EXITED 4 STATUS
```

Example 3

This example shows a run of the `TABLE9=TABLE9` in job `TestJob` with `-w` specified. The process exits with status 0 because the tables are in sync. Finished job statistics are displayed.

```
VERIDATA_DOMAIN_HOME\veridata\bin\vericom.bat -wuser veridata -wlpport 8830 -j
TestJob -g TestGroup -c TABLE9=TABLE9 -w
Connecting to: localhost:9177
Run ID: (2258, 0, 0)
Job Start Time: 2008-03-21 22:51:08
Job Stop Time: 2008-03-21 22:51:11
Job Report Filename: C:\veridata\data\rpt\TestJob\00002258\TestJob.rpt
Number of Compare Pairs: 1
Number of Compare Pairs With Errors: 0
Number of Compare Pairs With OOS: 0
Number of Compare Pairs With No OOS: 1
Number of Compare Pairs Cancelled: 0
Compare Pair Report Filename: C:\veridata\data\rpt\TestJob\00002258\TestGroup\CP_
TABLE9=TABLE9.rpt
Number of Rows Compared: 21
Number of Rows In Sync: 21
Number of Rows With Errors: 0
Number of Rows Out Of Sync: 0
Number of Inserts Out Of Sync: 0
Number of Deletes Out Of Sync: 0
Number of Updates Out Of Sync: 0
Compare Pair OOSXML Directory: C:\veridata\data\oosxml\TestJob\00002258\TestGroup
Compare Pair OOSXML Filename:
Job Completion Status: IN SYNC
C:\veridata\server\bin> if errorlevel 0 echo EXITED 0 STATUS
EXITED 0 STATUS
```

On UNIX systems, the exit status is in the '\$?' special variable if you use the SH or KSH shells. If you use the CSH shell, then the exit status is in the '\$status' special variable.

5

Using the Veridata Import and Export Utilities

In addition to using the Oracle GoldenGate Veridata Web User Interface, you can use the import and export utilities, provided with the Veridata installation, to define portions of your configuration.

This chapter includes the following sections:

5.1 Introduction to the Import and Export Utilities

Using the import and export 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 `ORACLE_HOME/veridata/clilib/lib/veridata-scripting.jar` file.

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 Veridata. Typically, the XML document matches the inputs on the configuration pages in the user interface.

The export utility helps you to either selectively or completely export the compare configuration data to an XML file. It can be used to export configurations from your current Veridata configuration or from other Veridata configurations using the `-repUrl` option. 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.

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

5.1.1 Supported Configurations

Oracle GoldenGate Veridata import and export utilities support configuring:

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

5.2 Running the Import and Export Utilities

The import and export utilities run from the `DOMAIN_HOME/veridata/bin` directory of the Oracle GoldenGate Veridata installation location. The Windows programs are

`veridata_export.bat` and `veridata_import.bat`; the UNIX and Linux scripts are `veridata_export.sh` and `veridata_import.sh`.

5.2.1 Using the Export Utility

The syntax for running the export utility is:

```
veridata_export[.sh | .bat] -export filePath -wuser commandlineUsername [-wport portNo] [-jobs jobName | -groups groupName | -connections connName | -profiles profileName | -all | -exportPassword] [[-repoUrl jdbc_url] [-u username>] [-schema schema_name] [-vdtPath VERIDATA_PRODUCT_HOME]]
```

- `-wport`: Represents the port for Veridata web server. The default value is 8830.
- `-wuser`: Specifies a user `commandlineUsername` with Veridata configuration privileges and command-line privileges.
- One of these optional operations can be requested at run time:
 - `-jobs`: Export all jobs, by name, including the associated groups, connections and profiles. You can specify one or more jobs by separating the names with a space, such as `job1 job2 job3`. If no job name is specified, all jobs with associated objects are exported.
 - `-groups`: Export all groups in the repository or add group names separated by a space, such as `group1 group2 group3`.
 - `-connections`: Export all connections in the repository or add connections separated by a space, such as `conn1 conn2 conn3`.
 - `-profiles`: Export all profiles in the repository or add profiles separated by a space, such as `profile1 profile2 profile3`.
 - `-all`: Export objects that are not part of any job. Takes precedence over all optional operations. This is the default when no other options are specified.
 - `-exportPassword`: Export the passwords for connections. By default, passwords for connections are not exported.
 - `-repoUrl`: Set the remote Veridata repository database JDBC URL for the export to use. You must set the `-u` option when using `-repoUrl`.

For Oracle Database:

```
jdbc:oracle:thin:@hostname/ip:port:SID
```

or

```
jdbc:oracle:thin:@hostname/ip:port/serviceName
```

For SQL Server:

```
jdbc:weblogic:sqlserver://hostname/ip:port;databaseName= databaseName
```

The `repoUrl` may require double quotes.

- `-u`: Set the remote Veridata database username from which the configuration export is requested. Use with the `-repoUrl` option.
- `-schema`: Set the remote Veridata schema name from which the configuration export is requested.

- `-vdtPath veridataLocation`: Set the Veridata domain location for the 12c release and later. For 11g release, it is the installation location.
- `-help`: Provides command line syntax.

If you want to export data from a Veridata repository database that is not part of your existing installation, you must provide the URL, username, and schema name. You will be prompted to enter the external Veridata repository password during run time. The Connection passwords are not exported by default, use the `-exportPassword` option for exporting passwords.

5.2.2 Using the Import Utility

The syntax for running the import utility is:

```
veridata_import[.sh | .bat] [-wlpport portNo] -wluser commandlineUsername [-create |
-update | -delete | -replace] configuration.xml
```

- `-wlpport`: Represents the port for Veridata web server. The default value is 8830.
- `-wluser`: Specifies a user *commandlineUsername* with Veridata configuration privileges and command-line privileges.
- One of these optional operations can be requested at run time:
 - `-create`: All items listed in the configuration are new. If any item in the list exists in the repository, nothing is added.
 - `-update`: 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.
 - `-delete`: All named items that exist for the configuration are removed from the repository.
 - `-replace`: All items listed to be replaced in the configuration are replaced as specified.
- *configuration.xml*: The name of the import XML configuration file that you created to describe the configuration. This is a required option.

5.2.3 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 `DOMAIN_HOME/veridata/logs/veridata_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.

5.2.4 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. Generate the keystore for the server and the client using the JDK keytool utility that is available at `JAVA_HOME/bin` as shown in the following sample script.

```
#####
# Creating Server and Client KeyStores.
#####
# NOTE: "keytool" is a utility that can be found inside the "$JAVA_HOME/bin" so
make sure the PATH includes the JDK's bin directory.

mkdir Keystores
cd Keystores/

echo ""
echo ""
echo "Creating WLS Server side Keystore. (wls12c.keystore)"
echo "-----"
keytool -genkey -v -alias wlsalias -keyalg RSA -keysize 1024 -keystore
wls12c.keystore -validity 3650 -keypass <keypassword> -storepass
<storepassword>+magic -dname "CN=127.0.0.1, OU=MiddlewareMagic, O=Blog,
L=Bangalore, S=Karnataka, C=IN"

echo ""
echo ""
echo "Exporting public key (wls12c_server.cer) from the WLS ServerSide keystore."
echo "-----"
keytool -export -keystore wls12c.keystore -alias wlsalias -file wls12c_server.cer -
keypass <keypassword> -storepass <storepassword>+magic

echo ""
echo ""
echo "Creating Client side Keystore/truststore. (clientTrustStore.keystore)"
echo "-----"
keytool -genkey -v -alias clientalias -keyalg RSA -keysize 1024 -keystore
clientTrustStore.keystore -validity 3650 -keypass <keypassword> -storepass
<storepassword> -dname "CN=127.0.0.1, OU=MiddlewareMagic, O=Blog, L=Bangalore,
S=Karnataka, C=IN"

echo ""
echo ""
echo "Importing the WLS Servers public key to the Client's truststore."
echo "-----"
```

```
keytool -import -v -trustcacerts -alias wlsalias -file wls12c_server.cer -  
keystore clientTrustStore.keystore -keypass <keypassword> -storepass  
<storepassword>
```

```
echo "Certificates created Successfully !!!"
```

2. Use the WebLogic Admin Console to setup the WebLogic Administrator Server and Oracle GoldenGate Veridata server keystore and SSL. For more information, see [Enabling End-to-End SSL](#).
3. Enter the absolute path location for the client keystore in the `veridata.cfg` configuration file as a key/value pair. For example:
`cli.ssl.client.trustStorePath=<absolute path for client keystore>.`
4. Navigate to the `VERIDATA_DOMAIN_HOME/veridata/bin` directory.
5. Run the Import utility in `ssl port` by passing the `-ssl` parameter as shown in the following example:

```
./veridata_import.sh -ssl -wlport <ssl_port> -wluser weblogic -  
create <configuration xml file>
```

6. Run the Export utility in `ssl port` by passing the `-ssl` parameter as shown in the following example:

```
./veridata_export.sh -export <export file name> -ssl -wlport <ssl  
port> -wluser weblogic -jobs <job name>
```

5.3 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`

[Sample Configuration File](#) provides a sample configuration file.

[Profile Parameters](#) provides a description of the profile parameters that you can use to configure your profiles.

5.4 configuration

The root element is `configuration`.

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

Table 5-1 configuration Elements

Elements	Description
<code>connection</code>	One or more Veridata database connection definitions.
<code>group</code>	One or more Veridata comparison group definitions.
<code>job</code>	One or more comparison job definitions.
<code>profile</code>	One or more profile definitions.

The following attributes describe the `configuration` element:

Table 5-2 configuration Attributes

Attribute	Description
<code>validation</code>	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 <i>default</i> 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.

Table 5-2 (Cont.) configuration Attributes

Attribute	Description
operation	<p>Specifies how data is applied to the repository. The options are:</p> <p>"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 <i>default</i> value.</p> <p>"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.</p> <p>"delete" - All named items in the configuration are removed from the repository.</p> <p>You can use a command line flag to override the value entered for this attribute.</p>
wildcard	<p>Specifies the pattern matching method that is used. The options are:</p> <p>"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.</p> <p>"regex" - Use regular expressions for matching.</p>

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>
```

5.5 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 5-3 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 <code>source-name</code> expression.

Table 5-3 (Cont.) column attributes

Attribute	Description
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".

5.6 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 5-4 colfilter Element

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

The following attribute describes the `colfilter` element:

Table 5-5 colfilter Attribute

Attribute	Description
type	Specifies whether to include the columns or exclude them. The options are <code>include</code> or <code>exclude</code> ; the default is <code>include</code> . 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>
```

5.7 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 5-6 colfiltercol Attribute

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

5.8 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 5-7 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.
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.

Table 5-7 (Cont.) compare-pair Elements

Element	Description
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 5-8 compare-pair Attributes

Attribute	Element
name	An expression defining the name of the compare pair. This expression can include groups captured with <code>source-table</code> expressions and target table group <code>\$0</code> .
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 <code>\$0</code> 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 <code>group</code> . 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 <code>group</code> . 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 <code>group</code> 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.
target-file-pattern	The default file pattern for the target if the data target is Enscribe or SQL/MP.

Table 5-8 (Cont.) compare-pair Attributes

Attribute	Element
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 <code>source-pkey</code> .
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 <code>key</code> , so the generated compare pair will have user-defined columns for the key. The options are: "true" - Compare pair has <code>key</code> column elements. This is the default. "false" - Compare pair does not have <code>key</code> column elements.
wildcard	Specifies the pattern matching method that is used. The options are: "ggg" - Use the typical Oracle GoldenGate pattern that matches an asterisk (*) to any number of characters. For Oracle GoldenGate HP Nonstop, the following is supported: <code>source-table=""target-table=""</code> , and the following is not supported: <code>source-table="TBL" target-table=""</code> . "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 <code>no-of-auto-partitions</code> . "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 <code>no-of-auto-partitions</code> .
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 5-8 (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>
```

5.9 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 5-9 connection Elements

Element	Description
<code>description</code>	Provides a description of the connection.
<code>conn-properties</code>	Defines the connection properties for a connection.

The following attributes describe the `connection` element:

Table 5-10 connection Attributes

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

```

5.10 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 5-11 conn-properties

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

5.11 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 5-12 delta-config Elements

Attribute	Description
<code>source-config</code>	Provides source side configuration for delta processing.
<code>target-config</code>	Provides target side configuration for delta processing.
<code>query</code>	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>
```

5.12 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>
```

5.13 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 5-13 `enscribe-info` Elements

Element	Description
<code>expandddl</code>	Describes the rules that are used when applying the DDL.

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

Table 5-14 `enscribe-info` Attributes

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

5.14 `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 be used in a where clause on the initial comparison query.

The following attributes describe the `enscribe-key`:

Table 5-15 `enscribe-key` Attributes

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

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"/>

```

5.15 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 5-16 excluded-column Attributes

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

5.16 expandddl

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

The following attributes describe the `expandddl` element:

Table 5-17 expandddl 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.

Table 5-17 (Cont.) expandddl Attributes

Attribute	Description
ddl-separator	The character separator for defining array output into columns. An example is the dash used in <code>FIELDX-3</code> , which is the third occurrence of <code>FIELDX</code> in the array. The options are: "none" - There is no separator. This is the <i>default</i> . "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.
fix-long-names	Whether to fix the names that result from resolving duplicates if they exceed the <code>max-col-name-length</code> . 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".

5.17 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 5-18 filter Attributes

Attribute	Description
type	Specifies either to include or exclude schemas and tables. Valid values are <code>include</code> or <code>exclude</code> .
catalog	Specifies the default catalog name.
exclude-views	Excludes all the views while generating compare pairs. Valid values are <code>true</code> or <code>false</code> . Default value is <code>false</code> .

Table 5-18 (Cont.) filter Attributes

Attribute	Description
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>
```

5.18 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 5-19 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 <code>compare-pair</code> 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 5-20 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.
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>

```

5.19 job

The `job` element defines an Oracle GoldenGate Veridata comparison job.

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

Table 5-21 job Elements

Element	Description
<code>description</code>	Provides a description of the job.
<code>group</code>	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 5-22 job Attributes

Attribute	Description
<code>name</code>	A name that identifies the job. This is a <i>required</i> attribute.
<code>source-conn</code>	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 <code>source-conn</code> is used to override the source connection specified for the groups included in the job.
<code>target-conn</code>	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.
<code>profile</code>	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>

```

5.20 profile

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

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

Table 5-23 profile Elements

Element	Description
<code>description</code>	Provides a description of the profile.
<code>profile-general</code>	Defines the profile parameters that control the output options.
<code>sorting-method</code>	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.
<code>initial-compare</code>	Defines the profile parameters that control the parameters for the job that performs the initial compare step
<code>confirm-out-of-sync</code>	Specifies the profile parameters that control the parameters for the job that performs the confirmation step
<code>repair</code>	Specifies the profile parameters that control the parameters for the repair job.

The following attributes describe the `profile` element:

Table 5-24 profile Attributes

Attribute	Description
<code>name</code>	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>
```

5.21 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 5-25 profile Attributes

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

5.22 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 5-26 profile-general Element

Element	Description
<code>param</code>	Defines the parameter to change for the profile.

5.23 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 5-27 sorting-method Element

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

5.24 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 5-28 initial-compare Element

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

5.25 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 5-29 confirm-out-of-sync Element

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

5.26 param

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

The following attributes describe the `param` element:

Table 5-30 param Attributes

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

5.27 repair

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

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

Table 5-31 repair Element

Element	Description
<code>param</code>	Defines the parameters that are used to configure the profile options.

5.28 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 5-32 sql-partition Attributes

Attribute	Description
<code>name</code>	A name that identifies the partition. This is a required attribute.
<code>side</code>	Indicates whether the partition should be applied at the source database, the target database, or both databases. The default is "both".
<code>default</code>	Indicates whether this is the default partition. This is equivalent to the "use at run time" indicator on the UI. The default is "false".
<code>type</code>	Distinguishes between Manual and Automatic Row Partition. When <code>type</code> is set to <code>sql</code> , it defines Manual Partition and <code>type</code> is set to <code>auto</code> , 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>
```

5.29 table partition

The `table-partition` element helps in specifying 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

Running Veridata GoldenGate Parameter Processing

Veridata GoldenGate Parameter Processing (VGPP) is a command-line tool that uses the map and table parameter of extract and replicat parameter files to create the Oracle GoldenGate Veridata configurations.

This chapter includes the following sections:

6.1 Overview of the Command-Line Interface

An Oracle GoldenGate Veridata parameter file contains all of the information required to extract or apply replicated data. The data-propagated replication is controlled by settings in the Extract and Replicat parameter files. Typically, the Extract parameter file specifies the tables to be replicated, the Replicat parameter file controls column mapping and restricts the tables. The Veridata GoldenGate Parameter Processing command-line utility accepts one or two parameter files as input. One of the files *must* be a Replicat parameter file, and the other optional file must be an Extract parameter file.

Replication captures information about the transaction responsible for changing the data as well as the actual changed data. However, Oracle GoldenGate Veridata can detect only the current state of the source and target databases, so it cannot support parameters for transactional changes, such as the `INSERTDELETES` parameter. Veridata GoldenGate Parameter Processing uses the MAP statements from a REPLICAT file to generate compare pairs. Other information in the parameter file is used to determine the relevant MAP statements. Optionally, you can use the TABLE statements from the EXTRACT parameter to restrict the compare pairs that are generated.

Because Oracle GoldenGate Veridata supports only single-column mapping, the column mapping assumes one-to-one mapping between source columns and target columns.

You can:

- Reuse Replicat and Extract configurations in Oracle GoldenGate Veridata
- Avoid creating separate Replicat and Extract parameter files.

6.2 Running the Veridata GoldenGate Parameter Processing

You can run the Veridata GoldenGate Parameter Processing program if you have the correct operating system permissions.

1. Ensure that the parameter files you want to use are on the system where the Oracle GoldenGate Veridata is installed. If you need to copy the files from another system, these files should be copied as binary files so that the `CHARSET` parameter remains valid.
2. Run the command shell of the operating system.
3. Navigate to the `VERIDATA_DOMAIN_HOME/veridata/bin` directory.
4. Run the Veridata GoldenGate Parameter Processing.

Syntax

```
veridata_param_process{.bat|.sh} required_parameter [optional_parameter]
```

Required Parameters

The following are required; otherwise an error is returned.

```
[-noscripting |
-create |
-replace |
-update ]
[-wlport port ] |
-wluser user_name |
[-p <propfile>]
[-o <outputFile>]
[replicat_param_filename]
```

The `-wluser` option specifies the Oracle GoldenGate Veridata Server (server) user name to connect to the server. This server user should have the `veridataCommandLineUser` privilege to access and execute command-line operations. The user should also have the `veridataAdministrator` or `veridataPowerUser` privilege to successfully run jobs and to use the import and export utilities.

Optional Parameters

The optional parameters are:

- [*extract_param_filename*]: If you provide the Extract and Replicat file names, then provide the Extract file name first, followed by the Replicat file name.

Table 6-1 VGPP Runtime Arguments

Argument	Description
{ <code>-noscripting</code> <code>-create</code> <code>-replace</code> <code>-update</code> }	Specifies that <code>veridata_scripting</code> is not run with the generated configuration file. The <code>-create</code> , <code>-replace</code> , and <code>-update</code> options indicate that <code>veridata_scripting</code> should be run with the generated configuration file. In either case, the generated scripting configuration file can be used as input to the <code>veridata_scripting</code> . The default option is <code>-create</code> .
<code>-wluser</code>	Specifies the Oracle WebLogic user name that authenticates and connects to the server.
<code>-wlport</code>	Specifies the Oracle WebLogic Server port number. The default listening port is 8830.
<code>-p</code>	Specifies a properties file containing additional information required for the configuration.
<code>-o</code>	Specifies the output file containing the generated scripting file. The default is <code>replicat_name_scripting.xml</code> ; <code>replicat_name</code> is the value of the <code>REPLCAT</code> parameter. The optional <code>extract_param_filename</code> parameter specifies an <code>EXTRACT</code> parameter file containing source information for the comparison. The <code>replicat_param_filename</code> parameter is the <code>REPLICAT</code> parameter containing the target information.

6.2.1 Using a Property File

When the VGPP program is run, an optional property file can be specified. This file contains information that is not available in the Oracle GoldenGate parameter file and is required to generate a valid Veridata comparison configuration. The following are some of properties (information) that you can specify.

Table 6-2 Optional Parameters

Property Name	Comments
<code>source.connection.name</code>	The name of the Veridata agent/manager connection. This may be the name of an existing Veridata connection. The default is the Extract name. This is the only source connection property needed to reference an existing connection.
<code>source.connection.port</code>	The port for the source agent. This is required when the connection does not already exist.
<code>source.connection.host</code>	The host name where the source agent is running. This is required when the connection does not exist.
<code>source.connection.user</code>	This defaults to the user information in the extract parameter file. This is required when the connection does not exist.
<code>source.connection.password</code>	This defaults to the user information in the extract parameter file. This is required when the connection does not already exist. If property name is specified without a value, the scripting utility will prompt for the value when the scripting configuration is loaded into Veridata.
<code>source.catalog</code>	This is valid for Sybase, SQL Server, and Oracle consolidated databases. For Sybase and SQL Server, it is the database containing the source tables. For Oracle, it specifies the Oracle PDB to use when processing an Extract parameter. Statements not associated with this PDB are ignored. The default value is the first PDB reference in the file. The reference can be a <code>SOURCECATALOG</code> parameter or the first part of a three-part name in a <code>TABLE</code> statement.
<code>extract.useansiquotes</code>	Indicates whether or not the Extract parameter file follows the ANSI quotation specification. This is a Boolean value. The default value is true. This is the same as the GoldenGate core <code>GLOBALS</code> parameters <code>USEANSISQLQUOTES</code> <code>NOUSEANSISQLQUOTES</code> .
<code>extract.charset</code>	The character set for the extract parameter file. This overrides any charset specified in the extract parameter file.
<code>extract.trail</code>	The trail file name to use when more than one trail file is specified in an extract parameter file. The default is the first trail file specified in the extract parameter file. When an extract parameter file contains multiple <code>rmttail</code> entries with the same name, Veridata will use the first occurrence.
<code>target.connection.name</code>	The name of the Veridata agent/manager connection. This may be the name of an existing Veridata connection. The default is the Replicat name. This is the only target connection property required to reference an existing connection.
<code>target.connection.port</code>	The port for the target agent. This is required when the connection does not already exist.

Table 6-2 (Cont.) Optional Parameters

Property Name	Comments
target.connection.host	The host name where the target agent is running. This is required when the connection does not exist.
target.connection.user	This defaults to the user information in the parameter file
target.connection.password	This defaults to the user information in the extract parameter file. If property name is specified without a value, the scripting utility prompts for the value when the scripting configuration is loaded into Veridata.
target.catalog	This is valid for Sybase and SQL Server. It is the database containing the target tables.
replicat.useansiquotes	Indicates whether or not the replicat parameter file follows the ANSI quotation specification. This is Boolean value. The default value is true. This is the same as the GoldenGate core GLOBALS parameters USEANSISQLQUOTES NOUSEANSISQLQUOTES .
replicat.charset	The character set for the replicat parameter file. This overrides any CHARSET specification in the replicat parameter file.

6.3 Parameter Handling

This section describes the handling of all of the parameters allowed in an Oracle GoldenGate Extract or Replicat parameter file. Each keyword is either supported, unsupported, or ignored. A supported parameter is used to generate the Veridata configuration. An unsupported parameter is something that interferes with the Veridata configuration generation. When an unsupported parameter (`INSERTDELETES`) is specified, subsequent `MAP` parameters are ignored. An ignored parameter specifies a feature that is not applicable to Veridata configuration generation.

The following table contains the known parameters and the expected handling: if a parameter is not listed, it is ignored.

Table 6-3 Parameter Handling

GoldenGate Parameters	Veridata Support
CATALOGEXCLUDE	This parameter is ignored. Veridata only processes items from a single catalog.
CHARMAP	Unsupported.
CHARSET	Supported. This parameter is supported in parameter files and include/obey files. Veridata does not process GLOBALS files.
COLMATCH	Supported.
COMMENT --	Supported.
DICTIONARY	Supported for NSK extract and replicat.
EXPANDDDL	Supported for NSK extract and replicat.
EXCLUDEWILDCARDOBJECTS ONLY	Supported.
EXTRACT	Supported.

Table 6-3 (Cont.) Parameter Handling

GoldenGate Parameters	Veridata Support
EXTTRAIL	Supported.
FILE TABLE	Supported for NSK extract.
INCLUDE	Supported. When the file is not found by the specified path, VGPP will look for the file name in the same directory as the parameter file.
INSERTALLRECORDS	Unsupported. Ignore all subsequent MAP statements.
INSERTDELETES NOINSERTDELETES	Unsupported Supported. Ignore all MAP statements following an INSERTDELETES command until a NOINSERTDELETES command is found.
INSERTUPDATES NOINSERTUPDATES	Unsupported Supported: Ignore all MAP statements between the INSERTUPDATES and the NOINSERTUPDATES.
MACRO	Supported.
MACROCHAR	Supported.
MAP	Supported.
MAPEXCLUDE	Supported.
OBEY	Supported. The same as INCLUDE.
REPLICAT	Supported.
RMTRAIL	Supported.
SCHEMAEXCLUDE	Supported.
SOURCECATALOG	Supported for Oracle consolidated databases.
TABLE MAP	Supported.
TABLEEXCLUDE	Supported.
UPDATEDELETES NOUPDATEDELETES	Unsupported Supported.
UPDATEINSERTS NOUPDATEINSERTS	Unsupported Supported.
USEANSISQLQUOTES NOUSEANSISQLQUOTES	Supported.

6.4 Map and Table Statement Handling

Oracle GoldenGate Veridata generates a compare pair element in the scripting configuration file for each Map statement in the Replicat parameter file. The generated scripting file lists the specific table mappings first, followed by the wildcard mappings, and finally the excluded mappings. This matches the behavior of the Oracle GoldenGate Replicat where specific mappings take precedence over wildcard mappings.

When the same source and target table specification appears in multiple MAP statements, the first occurrence will be used for the compare pair specification. The multiple occurrences can occur when the MAP statements use thread specifications and range filters.

The following table lists all of the keywords for the `MAP` and `TABLE` statements and support level in VGPP. `MAP` statements containing unsupported keywords will not generate a Oracle GoldenGate Veridata comparison configuration. Items marked with maybe indicate that more information is needed in order to determine the value for Oracle GoldenGate Veridata.

Table 6-4 Map and Table Statement Handling

Keyword	Veridata Support
TARGET	Supported.
COLMAP	Supported. Only simple source column to target column mapping is supported. Target columns mapped to functions or literals is excluded from the comparison configuration. The <code>USEDEFAULTS</code> keyword is supported. The <code>BINARYINPUT</code> keyword is ignored.
COLS	Supported. Results in an explicit column list in the generated compare pair configuration.
COLSEXCEPT	Supported. If an explicit column mapping does not exist, this results in system mapped columns with a list of omitted columns.
COMPARECOLS	Ignored
COORDINATED	Ignored.
DICTIONARY	Supported for NSK.
TARGETDICT	Supported for NSK.
DEF	Supported for NSK.
TARGETDEF	Supported for NSK.
EVENTACTIONS	Ignored.
EXCEPTIONSONLY	Unsupported.
EXITPARAM	Ignored.
FETCHBEFOREFILTER	Ignored
FETCHCOLS FETCHCOLSEXCEPT	Ignored
FETCHMODCOLS FETCHMODCOLSEXCEPT	Ignored
FILTER	Ignored.
GETBEFORECOLS	Ignored.
HANDLECOLLISIONS NOHANDLECOLLISIONS	Ignored
INSERTALLRECORDS	Unsupported.
INSERTAPPEND NOINSERTAPPEND	Ignored
KEYCOLS	Supported.
MAPEXCEPTION	Ignored.

Table 6-4 (Cont.) Map and Table Statement Handling

Keyword	Veridata Support
PARTITION	Supported. It defines the partition name for the table with or without wildcard symbol and considers only those partition for the compare pair.
PARTITIONEXCLUDE	Supported. It defines the partition name for the table with or without wildcard symbol to be excluded for the table. It also considers all other matched partition between source and target table partitions.
REPERROR	Ignored
RESOLVECONFLICT	Ignored.
SQLEXEC	Ignored
SQLPREDICATE	Ignored
THREAD	Ignored.
THREADRANGE	Ignored.
TOKENS	Ignored.
TRIMSPACES NOTRIMSPACES	Supported.
TRIMVARSPACES NOTRIMVARSPACES	Supported.
WHERE	Ignored.

6.5 VGPP Example

The example shows usage of VGPP utility using a simple extract file, replicat file, and property file.

```
./veridata_param_process.sh -noscripting -wuser veridata -wlport 8830 /scratch/ggcore/dirprm/extract.prm /scratch/ggcore/dirprm/replicat.prm -p properties.txt -o output.xml
```

The following is an example of including `PARTITION` and `PARTITIONEXCLUDE` parameters in the replicat files:

```
Replicat oraclesource
--SETENV (ORACLE_SID=ORA1110A)
USERID qatarget, PASSWORD qatarget
assumetargetdefs
discardfile ./dirrpt/robey.dsc, purge
map qasource.tcustmer, target qatarget.tcustmer;
MAP sales.order, TARGET rpt.*;
PARTITION sales.tab.part1;
PARTITION sales.order.part2;
PARTITION rpt.order.pa*;
PARTITIONEXCLUDE rpt.order.ts*;
```

Here: `extract.prm`: the extract param file. This is optional.

`replicat.prm`: the replicat param file. This is required.

`PARTITION sales.tab.part1`: includes Partition `part1` from Table `tab` and schema `sales`.

`PARTITIONEXCLUDE rpt.order.ts*`: excludes all partitions starting with `ts` from table `order` and schema `rp`.

If both the `extract` and `replicat` files are given, then the `extract` file should be given before `replicat` file.

`properties.txt`- this file contains properties related to veridata agents.

`output.xml`- this is the generated xml file which can be used with Veridata Scripting Import tool.

PARTITION - It defines the partition name for the table with or without wildcard symbol and considers only those partition for the compare pair.

PARTITIONEXCLUDE: It defines the partition name for the table with or without wildcard symbol to be excluded for the table. It also considers all other matched partition between source and target table partitions.

Sample outputs:

extract.prm

```
CHARSET utf-8
EXTRACT sqlParamSrcConnection
LOGALLSUPCOLS
RMTHOST localhost, MGRPORT 7000
RMTTRAIL sqlParamSrcConnection_trail
TABLE SOURCE.CHAR_*;
```

replicat.prm

```
CHARSET US-ASCII
REPLICAT sqlParamTrgConnection
ASSUMETARGETDEFS
MAP SOURCE.CHAR_TYPES, TARGET TARGET.CHAR_TYPES_NOTNULL, COLMAP ( USEDEFAULTS,
NCHAR_COL = NVARCHAR_COL, NVARCHAR_COL = NCHAR_COL, KEY_COL = CHAR_COL, CHAR_COL
= KEY_COL ),KEYCOLS (KEY_COL, CHAR_COL);
```

replicat.prm

```
Replicat oraclesource--SETENV (ORACLE_SID=ORA1110A)USERID
  qatarget, PASSWORD qatargetassumetargetdefsdiscardfile ./dirrpt/robey.dsc,
  purgemap qasource.tcustmer, target qatarget.tcustmer;MAP sales.order, TARGET
  rpt.*;PARTITION sales.tab.part1;PARTITION sales.order.part2;PARTITION
  rpt.order.pa*;PARTITIONEXCLUDE rpt.order.ts*;
```

replicat.prm, which has compare pairs generated with table partitions

```
<configuration wildcard="ggs" operation="create" validation="required">
<group name="oraclesource" source-conn="default_source_connection" target-
conn="oraclesource">
<compare-pair source-schema="qasource" source-table="tcustmer" target-
schema="qatarget" target-table="tcustmer"></compare-pair>
<compare-pair source-schema="sales" source-table="order" target-schema="rpt"
target-table="*">
<table-partition name="part2" default="true" side="source"></table-partition>
<table-partition name="pa*" default="true" side="target"></table-partition>
<table-partition name="ts*" default="true" side="target" exclude="true"></table-
partition>
</compare-pair>
</group>
<job name="oraclesource">
```

```
<group name="oraclesource"></group>
</job>
</configuration>
```

properties.txt

```
source.connection.host=localhost
source.connection.name=sqlParamSrcConnection
source.connection.port=7860
source.connection.user=source
source.connection.password=source
```

```
target.connection.host=localhost
target.connection.name=sqlParamTrgConnection
target.connection.port=7861
target.connection.user=target
target.connection.password=target
```

Generated output.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration SYSTEM "configuration.dtd">
<!--
Hostname : localhost
OGGV-30003: Extract filename : /scratch/ggcore/dirprm/extract.prm
OGGV-30002: Replicat filename : /scratch/ggcore/dirprm/replicat.prm
January 3, 2017 4:33:40 AM PST
-->
<configuration wilddcard="ggs" operation="create" validation="required">
  <connection name="sqlParamSrcConnection" host="localhost" port="7860" user="source"
password="source"></connection>
  <connection name="sqlParamTrgConnection" host="localhost" port="7861" user="target"
password="target"></connection>
  <group name="sqlParamSrcConnection_sqlParamTrgConnection" source-
conn="sqlParamSrcConnection" target-conn="sqlParamTrgConnection">
    <filter type="include" schema="SOURCE" table="CHAR_*"></filter>
    <compare-pair source-schema="SOURCE" source-table="CHAR_TYPES" target-
schema="TARGET" target-table="CHAR_TYPES_NOTNULL">
      <key-column target-name="KEY_COL" source-name="CHAR_COL"></key-column>
      <key-column target-name="CHAR_COL" source-name="KEY_COL"></key-column>
      <column source-name="*" type="hash"></column>
      <column target-name="NCHAR_COL" source-name="NVARCHAR_COL" type="hash"></
column>
      <column target-name="NVARCHAR_COL" source-name="NCHAR_COL" type="hash"></column>
    </compare-pair>
  </group>
</group>
<job name="sqlParamSrcConnection_sqlParamTrgConnection">
  <group name="sqlParamSrcConnection_sqlParamTrgConnection"></group>
</job>
</configuration>
```

7

Oracle GoldenGate Veridata Server Configuration 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:

7.1 Overview of the Server Memory

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

- The size of the memory mapped file, if using memory mapped file for sorting by setting the `server.mapped_sort_buffers` parameter to `true` in the `veridata.cfg` file.
- `java -Xmx` memory setting, if using java heap for sorting by setting the `server.mapped_sort_buffers` parameter to `false` in `veridata.cfg` file.

7.2 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.

The `server.max_sort_memory` of server parameter from the `veridata.cfg` file is used to calculate the maximum memory usage of the sorting. Therefore, ensure that the Java `-Xmx` value configured is larger than the sorting memory usage.

7.3 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.

7.4 Parameter Descriptions

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

7.4.1 Server Parameters

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

7.4.1.1 database.hash

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

Supported Values

true | false

Default

false



Note:

This parameter is currently supported for the Oracle database only.

7.4.1.2 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 = 0
```

Default Value

0

7.4.1.3 server.concurrent.readers

This parameter specifies the number of reader threads for the entire server.

Syntax

```
server.concurrent.readers number
```

Example

```
server.concurrent.readers number
```

Default Value

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

7.4.1.4 server.concurrent.writers

This parameter specifies the number of writer threads per sort directory.

Syntax

```
server.concurrent.writers number
```

Example

```
server.concurrent.writers number
```

Default Value

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

7.4.1.5 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. In this is not comparison format dependent, then one side should be Oracle and char, varchar, nchar, nvarchar, raw, clob, nclob, and blob datatypes of Oracle for which, this parameter is implicit. .

Syntax

```
server.convert_empty_to_null = false
```

Default

```
true
```

7.4.1.6 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.

7.4.1.7 server.max_concurrent_comparison_threads

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.

Syntax

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

- default : 100 threads

Example

```
server.max_concurrent_comparison_threads 100
```

Default Value

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

7.4.1.8 server.max_concurrent_jobs

This parameter specifies the maximum number of jobs that can be run simultaneously.

Syntax

```
server.max_concurrent_jobs number_of_jobs
```

Example

```
server.max_concurrent_jobs 200
```

Default Value

100

7.4.1.9 server.max_sort_memory

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`.

Syntax

```
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.

7.4.1.10 server.memory_mapped_sort_directory

This parameter specifies the directory path where `.map` files are generated. By default it uses temp directory of the operating system.

Syntax

```
server.memory_mapped_sort_directory directorypath
```

Example

```
server.memory_mapped_sort_directory directorypath/tmp2
```

7.4.1.11 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

7.4.1.12 server.persistence_db_type

This parameter defines the persistence database type.

Syntax

```
server.veridata_data database_type
```

where *database_type* is the persistence database type. The options are:

- ORACLE_OCI
- MS_SQL

Default Value

ORACLE_OCI

7.4.1.13 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
```

That means the default data directory is `VERIDATA_DOMAIN_HOME/veridata/reports`.

7.4.1.14 socket.timeout.mins

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

Default Value

```
0
```

7.4.1.15 truncate_spaces_len

Use the `truncate_spaces_len` parameter to control the length of truncated trailing spaces in a String or Binary column. 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 = " ".

Default

```
1
```

7.4.2 Parameters for Configuring SSL Communication

This section defines the parameters that you can use to configure SSL communication between your Oracle GoldenGate Veridata Server and Veridata Agents:

7.4.2.1 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.

7.4.2.2 server.ssl.client.allowTrustedExpiredCertificates

If the value of this parameter is set to true, Veridata Server allows SSL communication between the agent and the server when a trusted certificate expires.



Note:

The parameter is not applicable if you are running IBM's JVM.

Syntax

```
server.ssl.client.allowTrustedExpiredCertificates [true|false]
```

Example

```
server.ssl.client.allowTrustedExpiredCertificates false
```

Default Value

The default value is true.

7.4.2.3 server.ssl.client.identitystore.keyfactory.alg.name

This parameter specifies a name for the identity store key factory algorithm used for SSL communication.

Syntax

```
server.ssl.client.identitystore.keyfactory.alg.name=algorithm_name
```

Example

```
server.ssl.client.identitystore.keyfactory.alg.name=IbmX509
```

If you are running on IBM's JVM, set the value to IbmX509.

Default Value

The default value is SunX509.

7.4.2.4 server.ssl.client.truststore.keyfactory.alg.name

This parameter specifies a name for the trust store key factory algorithm used for SSL communication.

Syntax

```
server.ssl.client.truststore.keyfactory.alg.name=algorithm_name
```

Example

```
server.ssl.client.truststore.keyfactory.alg.name=IbmX509
```

If you are running on IBM's JVM, set the value to IbmX509.

Default Value

The default value is SunX509.

7.4.2.5 server.ssl.algorithm.name

This parameter specifies algorithm used for SSL communication.

Syntax

```
server.ssl.algorithm.name=algorithm_name
```

Example

```
server.ssl.algorithm.name=TLS
```

Default Value

The default value is TLS.

7.4.3 Parameters for Veridata Command-Line Utility

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

7.4.3.1 veridata.cli.run_from_managed_server

To run the Veridata command-line utility from the Veridata Managed Server, set this parameter value to `true`.

Syntax

```
veridata.cli.run_from_managed_server [true|false]
```

Example

```
veridata.cli.run_from_managed_server false
```

Default Value

The default value is true.

7.4.3.2 veridata.cli.server.listenAddress

This parameter specifies the listening address of the host machine for the Veridata Managed Server.

Syntax

```
veridata.cli.server.listenAddress host
```

Example

```
veridata.cli.server.listenAddress host.example.com
```

Default Value

The default name of the managed server is localhost.

7.4.3.3 veridata.cli.managed_server_name

This parameter specifies the name of the Veridata Managed Server.

Syntax

```
veridata.cli.managed_server_name server
```

Example

```
veridata.cli.managed_server_name VERIDATA_server2
```

Default Value

The default name of the managed server is VERIDATA_server1.

7.4.3.4 veridata.cli.server.timeout.seconds

This parameter specifies the time period (in seconds) Veridata CLI should wait for the JMX Server to respond to a CLI request.

Syntax

```
veridata.cli.server.timeout.seconds seconds
```

Example

```
veridata.cli.server.timeout.seconds 90
```

Default Value

The default time-out is 60 seconds.

7.4.4 Parameters for Report File Encryption

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

7.4.4.1 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.

7.4.4.2 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.

7.4.5 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:

7.4.5.1 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`.

7.4.5.2 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`.

7.4.5.3 mail.smtp.server

Syntax

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

Example

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

7.4.5.4 mail.smtp.port

Syntax

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

Example

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

7.4.5.5 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=false
```

Default Value

The default value is `false`.

7.4.5.6 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
```

7.4.5.7 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
```

7.4.6 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:

7.4.6.1 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
```

7.4.6.2 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

8

Agent Parameters

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

8.1 `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`.

Syntax

```
compare.xmldatatype.format=true
```

Default

```
true
```

8.2 `network.checksum.level`

Supported Value

```
REJECTED/ACCEPTED/REQUESTED/REQUIRED
```

Default Value

```
ACCEPTED
```

8.3 `network.checksum.types`

Default Value

```
SHA1
```

Example

```
network.checksum.types= SHA512
```

8.4 `network.encryption.level`

Supported Value

```
REJECTED/ACCEPTED/REQUESTED/REQUIRED
```

Default Value

```
ACCEPTED
```

8.5 network.encryption.types

This property is required to enable Network Encryption.

Example

```
network.checksum.types= SHA512
```

8.6 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.

8.7 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 and is useful only when the Delta column is set to `ORA_ROWSCN` which is the default delta column for all Oracle tables.

Syntax

```
rowscn=<scn number>
```

A

Moving from a Test to Production Environment

This appendix describes the scripts that we provide for you to migrate your test system to the production environment and includes the following sections:

A.1 Moving Installations from a Source Environment to a Target Environment

Oracle Fusion Middleware provides various scripts that you use to move your environment.

To move Oracle Home that contains all binary files of your Veridata installation, use the `copyBinary` and `pasteBinary` scripts. After moving the Oracle Home, use the `copyConfig` and `pasteConfig` scripts to move a copy of the Veridata domain configuration including the Administration Server, Managed Server, and other components in the domain to the target environment.



Note:

Test to production migration is not supported for a compact domain or for repository databases other than Oracle and SQL Server.

A.2 Additional Steps for Moving Oracle GoldenGate Veridata Repository

In addition to the common procedures described in the guide , follow the instructions below for moving the Oracle GoldenGate Veridata repository to a target environment:

A.2.1 Moving Veridata Configuration Data from Test to Production

To export and import Veridata repository configuration data, use the scripts available in the `DOMAIN_HOME/veridata/bin` directory use the following steps:

1. To export the repository from the test environment, run the export script as follows:

```
DOMAIN_HOME/veridata/bin/veridata_export.sh -export /tmp/export.xml -all -wuser  
cmd_user -exportPassword
```

[Table A-1](#) describes the arguments you specify while running the export and import scripts.

Table A-1 Arguments for the Veridata Test to Production Scripts

Argument	Description
export	Indicates that the script is exporting Veridata data from the test environment.
import	Indicates that the script is importing Veridata data to the production environment.
wluser <i>cmd_user</i>	Specifies the administrative WebLogic Server user.
exportPassword	Indicates that the script is exporting the <i>cmd_user</i> password.

2. Verify that any Veridata Agent host, port, and user password specifiers for the production host are updated in the just created `/tmp/export.xml` file.
3. If the Veridata Agent host or port has changed, then you must manually update the generated `export.xml` file with the new host and port informations for the Connections.
4. The export operation exports all of the data in the repository to XML file without validation. If your environment has some compare pairs for which the Validation Status is not successful, then the import operation will fail with the XML file generated by the export operation. To prevent failure of the import operation, you have the option to disable the compare pair validation. You can do this by updating the generated XML with these steps:

- a. Open the generated XML file.
- b. Search for the `configuration` tag. This tag has the `validation` attribute, which is set to `required`, by default.
- c. Set the `validation` attribute to one of the following values:

`omit-failures` — Indicates that all successfully validated compare pairs will be added to the repository and other specified compare pairs will be ignored.

`none` — Indicates that no compare pair validation is done before adding the compare pairs to the repository. You and then use the Veridata GUI to review and fix the validation problems. For example:

Old tag: `<configuration operation="update" validation="required">`

Updated tag: `<configuration operation="update" validation="omit-failures">`

5. To import the repository to the production environment, run the import script as follows:

```
DOMAIN_HOME/veridata/bin/veridata_import.sh -update /tmp/export.xml -
wluser cmd_user
```

A.2.2 Applying Configuration Changes while Moving from Test to Production

While moving from a test to production environment, if there are any configuration changes for the Veridata Agent such as host and port changes *or* if there is any schema or catalog name changes in the compare pairs, you must first execute the following statements:

Task 1 For all databases

```
Update DEV_VERIDATA.TABLE_INFO set SRC_SCHEMA_NAME = production_source_schema
Where SRC_SCHEMA_NAME = test_source_schema
Update TABLE_INFO set TARG_SCHEMA_NAME = production_target_schema Where
TARG_SCHEMA_NAME = test_target_schema
```

Where *DEV_VERIDATA* is the name of the production repository schema.

Where *production_source_schema* is the name of the production source schema and *production_target_schema* is the name of the production target schema.

Where *test_source_schema* is the name of the test source schema and *test_target_schema* is the name of the test target schema.

Task 2 Appropriate to your database

For SQL Server and Sybase databases:

```
Update TABLE_INFO set SRC_CATALOG_NAME = production_source_catalog Where
SRC_CATALOG_NAME = test_source_catalog
```

```
Update TABLE_INFO set TARG_CATALOG_NAME = production_target_catalog Where
TARG_CATALOG_NAME = test_target_catalog
```

Where *production_source_catalog* is the name of the production source catalog and *production_target_catalog* is the name of the production target catalog.

Where *test_source_catalog* is the name of the test source catalog and *test_target_catalog* is the name of the test target catalog.

For NSK:

Update the table names in the *COMPARE_PAIR* table to replace the test node names and disk volume names with the production names using one of the following appropriate for your database:

- **For Oracle:**

```
Update COMPARE_PAIRS SET SRC_TABLE_NAME = '\SPROD.$PDATA' ||
SUBSTR(SRC_TABLE_NAME, 12) Where SRC_TABLE_NAME like '\STEST.TDATA%'
Update COMPARE_PAIRS SET TARG_TABLE_NAME = '\TPROD.$PDATA' ||
SUBSTR(TARG_TABLE_NAME, 12) Where TARG_TABLE_NAME like '\TTEST.TDATA%'
```

- **For SQL Server:**

```
Update COMPARE_PAIRS SET SRC_TABLE_NAME = '\SPROD.$PDATA' +
SUBSTRING(SRC_TABLE_NAME, 12, LEN(SRC_TABLE_NAME) - 12) Where SRC_TABLE_
NAME like '\STEST.TDATA%'
Update COMPARE_PAIRS SET TARG_TABLE_NAME = '\TPROD.$PDATA' +
SUBSTRING(TARG_TABLE_NAME, 12, LEN(TARG_TABLE_NAME) -12) Where TARG_TABLE_
NAME like '\TTEST.TDATA%'
```

A.2.3 Modifying the Agent details in the Production Environment

Update the Veridata Agent details in the *CONNECTIONS* table of the production environment host as described below:

- If only the Agent host name needs to be changed, update the database as follows:

```
Update CONNECTIONS set MGR_NAME = 'prod host' where MGR_NAME 'test host'
```

- If there are more changes to the Veridata Agent, such as changes to the port number, User ID, password, and Repair User ID, then you should start the Veridata application and update the environment using the UI or command-line tool.

For example, create an /tmp/con.xml XML file as follows:

#Create xml as below by filling placeholders between @ and connections can be more than one.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration SYSTEM
"http://@VeridataServerHost@:@veridataServerPort@/veridata/
configuration.dtd">
<configuration validation="required">
<connection name="@conneiotnName@" host="@sqlManagerHostSrc@"
port="@sqlManagerPortSrc@" user="@sqlConn0User@"
password="@sqlConn0Password@" repairUser="@repairUsername@"
repairPassword="@repairPassword@" agent-timeout="4000"
truncate-spaces="false" fetch-size="3" use-ssl="false">
<description>
<![CDATA[
SQL Scripting Source Connection
]]>
</description>
</connection>
</configuration>
```

Update the Veridata Agent with your XML file using:

```
DOMAIN_HOME/veridata/bin/veridata_import.sh -update /tmp/con.xml -wlUser
cmd_user
```

Start the Veridata Agent after making these changes.

B

Sample Configuration File

For more information about the parameters used in this configuration file, see [Configuration File Element Reference](#).

B.1 Sample Configuration File

This section shows the contents of a sample configuration file. For more details about each element in this configuration file, see [Configuration File Element Reference](#).

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Copyright (c) 2011, Oracle and/or its affiliates. All rights reserved. -->
<!DOCTYPE configuration SYSTEM "configuration.dtd">
<configuration validation="required">
  <connection name="sqlScriptingSource" host="localhost" port="7860"
    user="source2" password="source2" agent-timeout="6000" truncate-spaces="true"
    fetch-size="4"/>
  <connection name="sqlScriptingTarget" host="localhost" port="7862"
    user="target2" password="target2"/>
  <connection name="connection-with-properties" host="localhost"
    port="7860" user="source2" password="source2" repairUser="source2"
    repairPassword="source2" agent-timeout="4000"
    truncate-spaces="false" fetch-size="3" use-ssl="false">
    <description>
      <![CDATA[ SQL Scripting Source Connection with user defined properties]]>
    </description>
    <conn-properties datatype-name="array" format="clob"/>
    <conn-properties datatype-name="binary_double" format="number" scale="3"/>
    <conn-properties datatype-name="binary_float" format="dec_float"
    precision="5"/>
    <conn-properties datatype-name="timestamp" format="binary_timestamp"
    scale="10" timezone="(UTC+05:30) Kolkata - India Time (IT)"/>
  </connection>
  <connection name="nskScriptingSource" host="gg-xxxx.us.company.com" port="9999"/>
  <connection name="nskScriptingTarget" host="gg-xxxx.us.company.com" port="9999" />

  <group name="column-mapping" source-conn="sqlScriptingSource" target-
    conn="sqlScriptingTarget"
    source-catalog="" target-catalog="" source-schema="SOURCE2" target-
    schema="TARGET2">
    <description>
      <![CDATA[
        This group has various types of column mapping specifications.
      ]]>
    </description>
    <compare-pair source-table="SYSMAPPING1" name="exlcudeCol6">
      <excluded-column name="COL6"/>
    </compare-pair>
    <compare-pair source-table="SYSMAPPING2" name="userDefinedKeys">
      <key-column source-name="COL1" target-name="COL2"/>
      <key-column source-name="COL2" target-name="COL3"/>
    </compare-pair>
  </group>
</configuration>
```



```

    <compare-pair source-table="SYSMAPPING3" name="userDefinedColsWildCard">
      <column source-name="COL.*" />
    </compare-pair>
    <compare-pair source-table="SYSMAPPING3" name="userDefinedColsLiteral">
      <column source-name="COL5" target-name="COL6" type="literal"/>
      <column source-name="COL.*" />
    </compare-pair>
  </group>

  <group name="table-mapping" source-conn="sqlScriptingSource" target-
conn="sqlScriptingTarget"
    source-catalog="" target-catalog="">
    <description>
      <![CDATA[
        This group has table mapping specifications.
      ]]>
    </description>
    <compare-pair source-table="SYSMAPPING1" target-table="SYSMAPPING1"
name="sameTables"
      source-schema="SOURCE2" target-schema="TARGET2" >
    </compare-pair>
    <compare-pair source-table="SYSMAPPING2" target-table="SYSMAPPING3"
name="diffTables"
      source-catalog="" target-catalog="" source-schema="SOURCE2" target-
schema="TARGET2">
    </compare-pair>
    <compare-pair source-table="CHAR_*" target-table=""
name="sameTables"
      source-schema="SOURCE2" target-schema="TARGET2" >
    </compare-pair>
  </group>

  <group name="delta-processing" source-conn="sqlScriptingSource" target-
conn="sqlScriptingTarget"
    source-catalog="" target-catalog="" source-schema="SOURCE2" target-
schema="TARGET2">
    <description>
      <![CDATA[
        This group has delta processing specifications.
      ]]>
    </description>
    <compare-pair source-table="SYSMAPPING1" target-table="SYSMAPPING1"
name="scriptingConfigTest1"
      delta-processing="true" >
      <key-column source-name="COL1" target-name="COL2"/>
      <key-column source-name="COL2" target-name="COL3"/>
      <column source-name="COL5" target-name="COL6" type="literal"/>
      <delta-config>
        <source-config column-name="COL1">
          <query><![CDATA[
            SELECT MAX(COL1) from SYSMAPPING1
          ]]>
        </query>
        </source-config>
        <target-config column-name="COL2">
          <query><![CDATA[
            SELECT MAX(COL1) from SYSMAPPING1
          ]]>
        </query>
        </target-config>
      </delta-config>

```

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

<group name="enscribe-partition" source-conn="SourceNSKConnection" target-
conn="TargetNSKConnection" validation="none">
  <description>
    <![CDATA[
      This group has all the tables for NSK
    ]]>
  </description>

  <compare-pair source-file-pattern="\ZEUS.$FSS02.FSSVSRC.*" source-
table="ACCTN*" target-file-pattern="\ZEUS.$FSS03.FSSVTAR.*" target-table="*" >
    <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"/>
  </compare-pair>
</group>

<group name="sql-partition" source-conn="sqlScriptingSource" target-
conn="sqlScriptingTarget"
  source-catalog="" target-catalog="" source-schema="SOURCE2" target-
schema="TARGET2">
  <description>
    <![CDATA[
      This group has sql partition specification.
    ]]>
  </description>

  <compare-pair source-table="SYSMAPPING1" name="PART1">
    <sql-partition name="partition_wo_default" >
      <![CDATA[
        col4 > 50
      ]]>
    </sql-partition>
    <sql-partition name="part2" side="source">
      <![CDATA[
        col2 > 20
      ]]>
    </sql-partition>
    <sql-partition name="part2" side="target">
      <![CDATA[
        col3 > 30
      ]]>
    </sql-partition>
  </compare-pair>
```

```

        <compare-pair source-table="SYSMAPPING2" name="PART2">
            <sql-partition name="partition_default" default="true" >
                <![CDATA[
                    col3 > 30
                ]]>
            </sql-partition>
        </compare-pair>
    </group>

    <group name="compare-pair-with-pkey" source-conn="sqlScriptingSource" target-
conn="sqlScriptingTarget"
        source-catalog="" target-catalog="" source-schema="SOURCE2" target-
schema="TARGET2">
        <description>
            <![CDATA[
                This group has all the SYSMAPPING tables.
                SYSMAPPING3 uses the user defined index B_SYSMAPPING4_IDX.
            ]]>
        </description>

        <compare-pair source-table="SYSMAPPING3" source-
pkey="B_SYSMAPPING3_IDX"/>
        <compare-pair source-table="SYSMAPPING*" target-table="*">
        </compare-pair>
        <compare-pair source-table="SYSMAPPING5" exclude="true"/>
    </group>

    <group name="enscribe-expand-ddl" source-conn="SourceNSKConnection" target-
conn="TargetNSKConnection" validation="none">
        <description>
            <![CDATA[
                This group has expand ddl specification for NSK
            ]]>
        </description>
        <compare-pair source-file-pattern="\ZEUS.$FSS02.FSSVSR.*" source-
table="TELLER" target-file-pattern="\ZEUS.$FSS03.FSSVTAR.*"
            target-table="TELLER" name="excludeCompKeyCol">
            <enscribe-info side="source"
                dictionary="$FSS04.FSSVDCT" record="ENSSRC-REC">
                <expandddl
                    redefined-columns ="include"
                    resolvedups="appendAlphaIndex"
                    expandGroupArrays="false"
                    ddl-separator="underscore"
                    zero-fill-length="1"
                    fix-long-names="false"
                    max-col-name-len="110"/>
            </enscribe-info>
            <enscribe-info side="target"
                dictionary="$FSS04.FSSVDCT" record="ENSSRC-REC">
                <expandddl />
            </enscribe-info>
            <excluded-column name="ENSCRIBE-NUMBER"/>
        </compare-pair>

        <compare-pair source-file-pattern="\ZEUS.$FSS02.FSSVSR.*" source-
table="TELLER" target-file-pattern="\ZEUS.$FSS03.FSSVTAR.*"
            target-table="TELLER" name="userDefined">
            <enscribe-info side="source"
                dictionary="$FSS04.FSSVDCT" record="ENSSRC-REC"/>
            <enscribe-info side="target"

```

```

        dictionary="$FSS04.FSSVDCT" record="ENSSRC-REC"/>
<key-column source-name="KEY1" target-name="KEY1"/>
<column source-name="ENSCRIBE-STRING" target-name="ENSCRIBE-STRING"/>
<column source-name="FIRST-NAME" target-name="FIRST-NAME"/>
<column source-name="LAST-NAME" target-name="LAST-NAME"/>
<column source-name="ENSCRIBE-NUMBER" target-name="ENSCRIBE-
NUMBER"/>
    </compare-pair>
</group>

<group name="include-exclude-filter" source-conn="sqlScriptingSource" target-
conn="sqlScriptingTarget"
    source-catalog="" target-catalog=""
    source-schema="SOURCE2" target-schema="TARGET2">
<description>
    <![CDATA[
        This group has include/exclude filter description
    ]]>
</description>

<filter type="include" table="SYSMAPPING*">
    <colfilter type="exclude">
        <colfiltercol name="COL3" />
        <colfiltercol name="COL6" />
    </colfilter>
</filter>
<filter type="exclude" table="SYSMAPPING4">
</filter>
<compare-pair source-table="SYSMAPPING1" target-table="*"
name="userDefinedCols"> <!-- exclude col6 -->
    <column source-name="COL5" target-name="COL5"/>
    <column source-name="COL6" target-name="COL6"/>
</compare-pair>
<compare-pair source-table="SYSMAPPING2" name="userDefinedKeys"> <!-- exclude
col3 -->
    <key-column source-name="COL1" target-name="COL2"/>
    <key-column source-name="COL2" target-name="COL3"/>
</compare-pair>
<compare-pair source-table="SYSMAPPING3" target-table="*"><!-- exclude col3,
col6 -->
</compare-pair>
<compare-pair source-table="SYSMAPPING4" target-table="*" />

</group>

<group name="quotedSchemaQuotedTable" source-conn="sqlScriptingSource" target-
conn="sqlScriptingTarget"
    source-catalog="" target-catalog=""
    source-schema=""abc 11"" target-schema=""abc 11"">
<description>
    <![CDATA[
        SQL group with simple quoted schema and quoted table name
    ]]>
</description>
<compare-pair source-table=""Quoted Table"" target-table="*" />
<compare-pair source-table=""Quoted*Table"" target-table="*" />
</group>

<group name="group-schema-wildcard" source-conn="sqlScriptingSource" target-
conn="sqlScriptingTarget"
    source-catalog="" target-catalog="" source-schema="source*" target-schema="*">

```

```

        <description>
            <![CDATA[
                SQL group with source-schema wildcard at group level and no
compare-pair schema.
            ]]>
        </description>
        <filter type="include" table="CHAR_TYPES*">
        </filter>
        <filter type="exclude" table="CHAR_TYPES2*">
        </filter>
        <compare-pair source-table="CHAR_TYPE*" target-table="">
        </compare-pair>
    </group>

<job name="test" profile="allParams">
    <group name="column-mapping"/>
    <group name="table-mapping"/>
    <group name="delta-processing"/>
</job>

<profile name="allParams">
    <description>
        <![CDATA[
            Full Profile description.
        ]]>
    </description>
    <profile-general>
        <param name="oos-format" value="xml" />
        <param name="oos-xml-chunk-size" value="1000" />
        <param name="reports-insync" value="true" />
        <param name="reports-inflight" value="true" />
    </profile-general>
    <sorting-method>
        <param name="sort-method" value="server" />
        <param name="sort-src-temp-dir" value="/dummy/location" />
        <param name="sort-tar-temp-dir" value="/dummy/location" />
    </sorting-method>
    <initial-compare>
        <param name="max-thread" value="6" />
        <param name="max-oos-record" value="777777" />
        <param name="output-oos-rpt" value="true" />
        <param name="update-rpt-second" value="100" />
        <param name="update-rpt-record" value="100" />
        <param name="limit-input-row" value="100" />
        <param name="src-oracle-hint" value="FIRST_ROWS(10)" />
        <param name="tar-oracle-hint" value="FIRST_ROWS(10)" />

        <param name="rpt-msg" value="both" />
        <param name="rpt-warn-msg-threshold" value="100" />

        <param name="src-agent-static-port" value="777" />
        <param name="tar-agent-static-port" value="777" />

        <param name="src-nsk-name" value="$AA*" />
        <param name="src-nsk-cpu" value="2" />
        <param name="src-nsk-priority" value="1" />
        <param name="tar-nsk-name" value="$AA*" />
        <param name="tar-nsk-cpu" value="2" />
        <param name="tar-nsk-priority" value="1" />
    </initial-compare>

```

```
<confirm-out-of-sync>
  <param name="coos-enable" value="false" />
  <param name="coos-concurrent" value="false" />
  <param name="batch-size" value="15"/>
  <param name="coos-delay" value="2" />
  <param name="max-oos-record" value="777777" />
  <param name="output-oos-rpt" value="true" />
  <param name="update-rpt-second" value="100" />
  <param name="update-rpt-record" value="100" />
  <param name="src-oracle-hint" value="FIRST_ROWS(10)" />
  <param name="tar-oracle-hint" value="FIRST_ROWS(10)" />

  <param name="rpt-msg" value="both" />
  <param name="rpt-warn-msg-threshold" value="100" />

  <param name="src-agent-static-port" value="777" />
  <param name="tar-agent-static-port" value="777" />

  <param name="src-nsk-name" value="$AA*" />
  <param name="src-nsk-cpu" value="2" />
  <param name="src-nsk-priority" value="1" />
  <param name="tar-nsk-name" value="$AA*" />
  <param name="tar-nsk-cpu" value="2" />
  <param name="tar-nsk-priority" value="1" />
</confirm-out-of-sync>
<repair>
  <param name="repair-after-compare" value="true" />
  <param name="batch-size" value="15" />
  <param name="txn-size" value="2" />
  <param name="concurrent-operation" value="2" />
  <param name="check-change-value" value="false" />
  <param name="terminate-max-warn" value="77777" />
  <param name="write-success-rpt" value="false" />
  <param name="disable-trigger" value="true" />
</repair>
</profile>
</configuration>
```

C

Profile Parameters

This appendix describes how to configure the profiles used with the Oracle GoldenGate Veridata import and export utilities.

For more information about the parameters used in this configuration file, see [Configuration File Element Reference](#).

C.1 General (profile-general)

Name on UI	Name to be used in configuration XML file	Default Values	Allowed Values	Type
Out-Of-Sync Output Format	oos-format	binary	binary, xml, both, none	Enumeration
Maximum Size of Each Out-Of-Sync XML Chunk (Rows)	oos-xml-chunk-size	500	1 to 100000	int
Report in-sync rows to report file	reports-insync	false	true, false	boolean
Report in-sync after in-flight rows to report file	reports-inflight	false	true, false	boolean

C.2 Sorting Method (sorting-method)

Name on UI	Name to be used in configuration XML file	Default Values	Allowed Values	Type
Sort Data Using	sort-method	database	server, database	Enumeration
Temporary Storage Directory for Source Data	sort-src-temp-dir	string	NA	NA
Temporary Storage Directory for Target Data	sort-tar-temp-dir	string	NA	NA

C.3 Initial Compare (initial-compare)

Name on UI	Name to be used in config XML file	Default Values	Allowed Values	Type
Max Concurrent Comparison Threads	max-thread	4	1 to 20	Int
Terminate when Maximum Records Out-Of-Sync	max-oos-record	100000	0 to 100000000	int
Output Out-Of-Sync Record Details to Report File	output-oos-rpt	false	true, false	boolean
Update Report file Every (seconds)	update-rpt-second	0	0 to 1000000	int
Update Report file Every (records)	update-rpt-record	0	0 to 100000000	int
Limit Number of Input Rows	limit-input-row	0	0 to 100000000	int
Source Oracle optimizer hint	src-oracle-hint	string	NA	NA
Target Oracle optimizer hint	tar-oracle-hint	string	NA	NA
Generate Messages	rpt-msg	none	none, info, warning, both	Enumeration
Generate Warning Messages For Out-Of-Sync Rows After (differences)	rpt-warn-msg-threshold	50	0 to 1000000	int
Use Static Listening Port For Agent During Row Hash On Source (0 to use dynamic port list)	src-agent-static-port	0	0 to 65535	int
Use Static Listening Port For Agent During Row Hash On Target (0 to use dynamic port list)	tar-agent-static-port	0	0 to 65535	int

Name on UI	Name to be used in config XML file	Default Values	Allowed Values	Type
Source Process Name Starting With (Must start with '\$', followed by two letters, and end with '*'. Example: \$AA*)	src-nsk-name	string	NA	NA
Source Process CPU Number	src-nsk-cpu	-1	-1 to 16	int
Source Process Priority	src-nsk-priority	0	0 to 1000000	int
Target Process Name Starting With (Must start with '\$', followed by two letters, and end with '*'. Example: \$AA*)	tar-nsk-name	string	NA	NA
Target Process CPU Number	tar-nsk-cpu	-1	-1 to 16	int
Target Process Priority	tar-nsk-priority	0	0 to 1000000	int

C.4 Confirm-Out-Of-Sync (confirm-out-of-sync)

Name on UI	Name to be used in configuration XML file	Default Values	Allowed Values	Type
Perform Confirm Out-Of-Sync Step By default, Perform Confirm Out-Of-Sync Step is enabled. You can make Initial Compare Terminate when Maximum Records Out-Of-Sync effective ONLY when Perform Confirm Out-Of-Sync Step is disabled. This way, the MOOS will include all OOS and not subject to 100k limit when COOS is enabled.	coos-enable	true	true, false	boolean

Name on UI	Name to be used in configuration XML file	Default Values	Allowed Values	Type
Run Concurrently With Initial Compare	coos-concurrent	true	true, false	boolean
Confirm-Out-Of-Sync Batch Size	Batch-size	10	1 to 100	int
Delay Confirm-Out-Of-Sync By (seconds)	coos-delay	0	0 to 1000000	int
Terminate when Maximum Records Out-Of-Sync	coos-max-oos	100000	0 to 100000000	int
Output Out-Of-Sync Record Details to Report File	coos-output-oos-rpt	false	true, false	boolean
Update Report file Every (seconds)	coos-output-rpt-second	0	0 to 100000000	int
Update Report file Every (records)	coos-output-rpt-record	0	0 to 100000000	int
Source Oracle optimizer hint	coos-src-oracle-hint	string	NA	NA
Target Oracle optimizer hint	coos-tar-oracle-hint	string	NA	NA
Generate Messages	rpt-msg	none	none, info, warning, both	Enumeration
Generate Warning Messages For Out-Of-Sync Rows After (differences)	rpt-warn-msg-threshold	50	0 to 1000000	int
Use Static Listening Port For Agent During Row Hash On Source (0 to use dynamic port list)	src-agent-static-port	0	0 to 65535	int
Use Static Listening Port For Agent During Row Hash On Target (0 to use dynamic port list)	tar-agent-static-port	0	0 to 65535	int
Source Process Name Starting With (Must start with '\$', followed by two letters, and end with '*'. Example: \$AA*)	src-nsk-name	string	NA	NA

Name on UI	Name to be used in configuration XML file	Default Values	Allowed Values	Type
Source Process CPU Number	src-nsk-cpu	-1	-1 to 16	int
Source Process Priority	src-nsk-priority	0	0 to 1000000	int
Target Process Name Starting With (Must start with '\$', followed by two letters, and end with '*'. Example: \$AA*)	tar-nsk-name	string	NA	NA
Target Process CPU Number	tar-nsk-cpu	-1	-1 to 16	int
Target Process Priority	tar-nsk-priority	0	0 to 1000000	int

C.5 Repair (repair)

Name on UI	Name to be used in configuration XML file	Default Values	Allowed Values	Type
Run Repair Automatically after Compare	repair-after-compare	False	True,false	boolean
Repair batch size	batch-size	10	1 to 100	int
Repair transaction size	txn-size	1	0 to 100	int
Concurrent Repair Operations	concurrent-operation	1	1 to 100	int
Check changed values	check-change-value	true	true, false	boolean
Terminate when maximum repair warnings	terminate-max-warn	10000	0 to 2147483647	Int
Write Repair Success Messages to Report	write-success-rpt	true	true, false	boolean
Disable DB Triggers Session Based Report	disable-trigger	false	true, false	boolean
Disable DB Triggers Session Based	disable-trigger	false	true, false	boolean