Oracle® GoldenGate

Administering Oracle GoldenGate Veridata

12*c* (12.1.3)

E29293-02

June 2015

This guide explains how to run and administer the Oracle GoldenGate Veridata data comparison solution.

ORACLE®

Oracle GoldenGate Administering Oracle GoldenGate Veridata, 12*c* (12.1.3)

E29293-02

Copyright © 2005, 2015, Oracle and/or its affiliates. All rights reserved.

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 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 installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. 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 and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon 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, Opteron, the AMD logo, and the AMD Opteron 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

4 Running Comparisons from the Command Line

5 Configuration Scripting

6 Oracle GoldenGate Veridata Server Configuration Parameters

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.

Access to Oracle Support

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

Related Documents

The complete Oracle GoldenGate documentation set includes the following components:

- *Release Notes for Oracle GoldenGate Veridata*
- *Installing and Configuring Oracle GoldenGate Veridata*
- *Upgrading Oracle GoldenGate Veridata*
- *User's Guide*

Conventions

The following text conventions are used in this document:

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

What's New

This chapter summarizes the new features and significant product changes for Oracle GoldenGate Veridata in Oracle GoldenGate Veridata 12c (12.1.3) release.

This chapter includes the following sections:

New and Changed Features for Release 12*c* (12.1.3)

Oracle GoldenGate Veridata Release 12c (12.1.3) includes the following new and changed features.

- New heterogeneous Repair feature, which fixes out-of-sync data in a real-time environment. See "Repairing out-of-sync Jobs" in *Oracle GoldenGate Veridata User's Guide*.
- New Scripting tool, which allows you to load comparison configuration from an XML file. See Chapter 5, "Configuration Scripting".
- IPv6 support for Java Agents.
- Purging, which helps you remove unwanted job reports and data. See *Oracle GoldenGate Veridata Online Help*.
- Support for Single Sign-on. See Section 1.2, "Configuring Single Sign-on for Oracle GoldenGate Veridata".
- Security enhancements:
  - Support for SSL Communication. See Section 2.2, "Configuring an SSL Connection between Veridata Server and Veridata Agents".
  - Added new security roles to Oracle GoldenGate Veridata users. See Section 2.4, "Securing Access to Oracle GoldenGate Veridata by Defining User Roles".
  - Report file encryption. See Section 2.5, "Encrypting Report Files".
- New installation process. From this release, use Oracle Universal Installer to install and configure a Oracle WebLogic Server domain that hosts the Veridata Server. See *Installing and Configuring Oracle GoldenGate Veridata*.

Other Significant Changes in this Document for Release 12c (12.1.3)

For 12c (12.1.3), this guide has been updated in several ways. Following are the sections that have been added or changed.

- Added a chapter that describes how to use the scripting tool. See Chapter 5, "Configuration Scripting".
- Added a section that describes how to configure SSL for Oracle GoldenGate Veridata. See Section 2.2, "Configuring an SSL Connection between Veridata Server and Veridata Agents".
- Added new command-line options and descriptions. See Chapter 3, "Running the Oracle GoldenGate Veridata Programs".

1

Introduction to Oracle GoldenGate Veridata

This chapter describes how to use Oracle GoldenGate Veridata. It provides an overview of roles and interactions of the components, how to configure components, and how Oracle GoldenGate Veridata compares tables and repairs out-of-sync tables.

This chapter includes the following sections:

1.1 Oracle GoldenGate Veridata Architecture

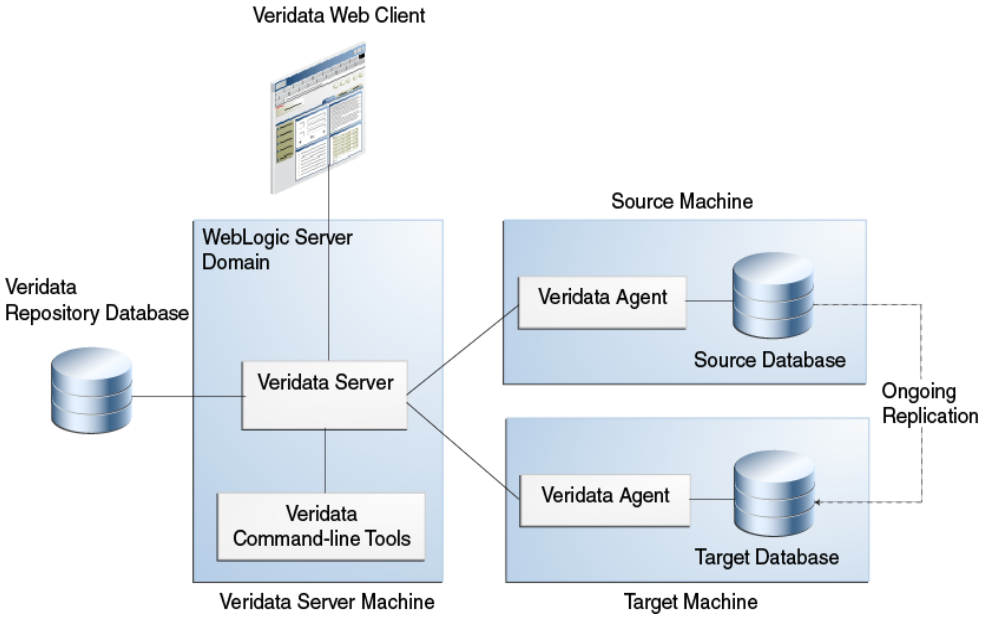
Oracle GoldenGate Veridata compares one set of data to another and identifies data that is out-of-sync, and allows you to repair any data that is found out-of-sync. Oracle GoldenGate Veridata supports high-volume, 24x7 heterogenous replication environments where downtime to compare data sets is not an 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, while still producing an accurate comparison report.

Oracle GoldenGate Veridata will map column data types across different types of databases automatically, or you can map columns manually in cases where the automatic mapping is not sufficient to accommodate format differences in a heterogeneous environment. For detailed information about this feature in Veridata Web User Interface, see *Oracle GoldenGate Veridata Online Help*. Alternatively, you can map data by manually uploading an XML file using the `veridata_scripting` utility. For more information, see Chapter 5, "Configuration Scripting".

For the purposes of this documentation, the following terms are considered synonymous:

- tables and files
- columns and fields
- rows and records

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 (optional)
- Compare data
- Confirm out-of-sync data
- Produce a report for review

Oracle GoldenGate Veridata Web User Interface

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 Veridata repository. This data is stored in files on the file system of the Veridata Server.

Oracle GoldenGate Veridata Agent

The Oracle GoldenGate Veridata 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 detail for out-of-sync rows

Oracle GoldenGate Veridata Manager

The Manager process is part of the C-code based agent that is required for the NonStop platform. It controls the Oracle GoldenGate Veridata 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 Chapter 4, "Running Comparisons from the Command Line".
`veridata_scripting`	Maps comparison objects and rules in an XML file and uploads into the repository. See Chapter 5, "Configuration Scripting"
`reportutil`	Supports viewing encrypted report files and out-of-sync data.

1.2 Configuring Single Sign-on for Oracle GoldenGate Veridata

Oracle GoldenGate Veridata 12.1.3 supports Single Sign-on (SSO) mechanism for authentication. To configure SSO, you should set the SSO properties for Veridata Server and also configure the logout URL for the SSO session.

The web parameter `web.singleSignOutUrl` in the `DOMAIN_HOME/config/veridata/veridata.cfg` file is used to configure SSO for the Veridata Server. The parameter usage is explained in the `veridata.cfg` file:

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

To configure SSO, set the `web.singleSignOutUrl` parameter and run the `configureVeridata` script as follows:

To Configure SSO Logout

```
DOMAIN_HOME/veridata/bin/configureVeridata.sh -pUweb.singleSignOutUrl=Single sign out URL
```

To Reset SSO Logout

If your domain is no longer using SSO, you can optionally remove the SSO logout configuration as follows:

```
DOMAIN_HOME/veridata/bin/configureVeridata.sh -pUweb.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 need to create some objects that help you identify the data you want to compare and help you manage your work.

- Configure data source connections: Oracle GoldenGate Veridata Server connects to Oracle GoldenGate Veridata Agents that interact with the databases that contain the data that is to be compared. A connection is defined by a host, the port number of an Oracle GoldenGate Veridata Agent on that host, and the data source that is accessed by the agent.
- Configure groups: You need to configure at least one compare group that is linked to a set of source and target data source connections. A group is a logical container for organizing the objects to be compared.
- Configure compare pairs: You need to configure one or more compare pairs for each group that you create. A compare pair is a set of corresponding source and target tables or files. As part of configuring compare pairs, you map source and target columns to establish a structural relationship between the two objects.
- Configure profiles: A profile contains settings for run-time parameters and can be applied globally to a job, as well as to a specific compare pair as an override to the job profile. Profile parameters encompass considerations such as sorting method, thread and memory usage, reporting output, and so forth. Defining run profiles is optional, because Oracle GoldenGate Veridata includes a default profile that contains settings that apply to most usage scenarios. However, as you gain experience with Oracle GoldenGate Veridata, you may want to customize the default profile or create your own custom profiles.
- Configure jobs: A job is a logical container for one or more compare groups and is the unit of work by which comparison processing is executed. Within one or more jobs, you can manage and run large volumes of compare groups across numerous databases and systems, and you can control the timing of those comparisons.

For more information, see the *Oracle GoldenGate Veridata Online Help*.

1.3.2 Satisfying Uniqueness Requirements

Oracle GoldenGate Veridata relies on some form of 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, Oracle GoldenGate Veridata uses the smallest unique index
- **User-defined Key:** If a table or file has neither a primary nor unique key, you can define an existing index or set of columns for comparison purposes when defining a compare pair. However, although primary or unique keys can be mapped automatically, user-defined keys must be mapped manually. A user-defined key can also be used to override existing keys or indexes if you prefer a different ordering method.

For more information about choosing and mapping keys for comparison, see the Oracle GoldenGate Veridata online help.

1.3.3 How Oracle GoldenGate Veridata Compares Data

Comparison activities are divided into the following steps. You can change some of the aspects of these steps by making parameter changes in Oracle GoldenGate Veridata Web.

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, while still providing a highly reliable (but not absolute) and efficient mechanism for determining whether two rows contain the same or different column values.

For more assurance of discovering out-of-sync rows, you can configure Oracle GoldenGate Veridata to compare non-key rows column-for-column, instead of using a hash. Full-column comparisons reduce the processing performance in proportion to the number of columns, and they increase network usage.

On the NonStop platform, you can use the delta processing feature during the initial comparison step if you are using server-side sorting. Delta processing is a performance feature by which Oracle GoldenGate Veridata detects which data blocks in the database 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 regardless of whether they changed or not.

There are two steps to delta processing:

- Collecting the base modification time of the previous run for subsequent delta comparisons. This step is always done when delta processing is enabled for a compare pair.
- Comparing data that has been modified since the base comparison, using the information that was collected in the first step. This step is enabled by clicking the **Enable Delta Processing** button on the Compare Pair Configuration page and the Run/Execute Job page of Oracle GoldenGate Veridata Web. The **Disable Delta Processing** button allows you to disable the delta comparison step in case there were modifications, such as table reorganizations, that can invalidate the collected delta base information.

For more information about delta processing, see the *Oracle GoldenGate Veridata Online Help*.

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 at this point the comparison is inconclusive. When replication is working concurrently with a comparison, especially if there is replication latency, rows can appear to be out-of-sync when, in fact, the current data is *in flight* (somewhere in the replication flow) and replication will soon synchronize them again.

1.3.3.2 Confirmation Step

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

- *in-flight*: the row was out-of-sync in the initial comparison step, but has since been updated. In this case, 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. Even a status of in-sync 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 has not been updated since the initial comparison step took place, and therefore 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 after a specified replication latency threshold has expired. 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

Upon completion of a job, you can view the comparison reports and the out-of-sync report by using Oracle GoldenGate Veridata Web User Interface or by viewing the files themselves.

If report encryption is enabled for the Veridata Server, you need to use the `reportutil` tool to view the report files. See Section 2.5, "Encrypting Report Files". The Veridata Web User Interface automatically decrypts the file before displaying them.

1.4.1 Out-of-Sync Report

You have the option to store an out-of-sync 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 are used for viewing row differences using the Oracle GoldenGate Veridata Web Interface, and the report is also used to re-compare out-of-sync rows later. To re-compare rows, you select run options to execute another confirmation step, which compares the current state of just those rows and then reports which ones remain out-of-sync after replication or another restorative procedure has been applied.

- **OOSXML file**: When stored as XML, the OOS report is written to an `OOSXML` file and is stored in a structured way that conforms to an internal XML schema. XML has many advantages, the largest being that it can be manipulated easily by many tools. In its XML form, the file contains all of the information, including metadata, that is needed to select rows for re-synchronization by external programs.

1.4.2 Comparison Report

Each finished job, group, and compare pair generates a comparison report. The report file contains details about the comparisons that were performed, such as:

- Comparison parameters used
- The number of rows compared and out-of-sync
- The timing of the comparison
- Performance statistics
- Source and target data values

The files themselves are stored as follows:

By default, the OOS files are located in sub-directories of the Oracle GoldenGate Veridata Server installation directory:

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

You can change the default location by specifying another path for the server.veridata_data property in the `veridata.cfg` file.

These directories are further organized by run ID, job name, group name, and compare pair. In the `OOSXML` directory, the files with the `.oosxml` extension are the control files. The files with sequential file extensions are the `OOSXML` chunks. The XML data is spread into multiple files (called "chunks") for performance purposes.

You can choose to encrypt the comparison reports. For more information, see Section 2.5, "Encrypting Report Files".

2

Configuring Oracle GoldenGate Veridata Security

This chapter explains how to set security for Oracle GoldenGate Veridata.

This chapter includes the following sections:

2.1 Oracle GoldenGate Veridata Security Overview

When using Oracle GoldenGate Veridata, you will be selecting, viewing and storing data values from the tables or files of your business applications. Care must be taken to protect access to the following components:

- The files, programs, and directories in the Oracle GoldenGate Veridata installation directories
- Data files that contain the results of data comparisons
- The Oracle GoldenGate Veridata Web User Interface, where data values can be viewed

2.2 Configuring an SSL Connection between Veridata Server and Veridata Agents

Oracle GoldenGate Veridata supports both Secure Sockets Layer (SSL) and plain socket communication between the Veridata Server and multiple Veridata Agents that are connected over a network. This section describes how to configure SSL and secure communication between the Veridata Server and Veridata Agents.

Note: The Veridata Agent for NonStop platforms do not support SSL communication.

In an SSL scenario, the Veridata Server is considered as the SSL Client and the Veridata Agents as the SSL Servers. The Veridata Server and Agents authenticate each other's identity. The data exchanged between the server and agent is also encrypted.

2.2.1 One-Way and Two-Way SSL Connections

SSL can be configured one-way or two-way in Oracle GoldenGate Veridata.

In one-way SSL connection, the SSL Client (Veridata Server) should trust the SSL Server (Veridata Agent). In two-way SSL, mutual trust is required between the SSL Server and the SSL Client. You can either use self-signed certificates or CA signed certificates to enable SSL.

Using self-signed certificates

To establish one-way SSL using self-signed certificates:

- Create self-signed certificates for all Veridata Agents
- Upload all Veridata Agent certificates to the `VeridataWebTrustStore` of the Veridata Server. See Section 2.2.6, "Using OPSS Keystore Service to Manage Veridata Keystores".

To establish two-way SSL using self-signed certificates:

- Create self-signed certificates for all Veridata Agents.
- Upload all Veridata Agent certificates to the `VeridataWebTrustStore` of the Veridata Server. See Section 2.2.6, "Using OPSS Keystore Service to Manage Veridata Keystores".
- Create self-signed certificate for the identity store of the Veridata WebLogic Server.
- Upload the WebLogic Server identity certificate to all Veridata Agent truststores.

For more information about creating and importing certificates, see Section 2.2.5, "Creating Keystores and Self-Signed Certificates by using the Keytool Utility".

Using CA-signed certificates

To establish one-way SSL using CA signed certificates:

- Use certificates issued by the same Certificate Authority (CA) for all Veridata Agents.
- Trust the root CA certificate in the Veridata Weblogic Server.

To establish two-way SSL using CA signed certificates:

- Use certificates issued by the same Certificate Authority (CA) for all Veridata Agents.
- Trust the root CA certificate in the Veridata Weblogic Server.
- Use the certificate issued by a CA for identity store of Veridata Weblogic Server.
- Trust the root CA certificate used in the previous step in the Veridata agent truststore.

2.2.2 Enabling SSL: Main Steps

Oracle GoldenGate Veridata Server and Oracle GoldenGate Veridata Agents are not enabled for SSL by default. If you decide to use SSL, you must enable the properties for the server and the agents. See SSL Settings for Oracle GoldenGate Veridata Agent and SSL Settings for GoldenGate Veridata Server.

You must also create the identity and trust keystores. Create self-signed certificates if you are not using a Certificate Authority (CA) certificate. See Creating Keystores and Self-Signed Certificates by using the Keytool Utility.

To verify and establish an SSL connection between the Veridata Server and an Agent, follow these steps:

1. Configure SSL for the Veridata Web Server. See SSL Settings for GoldenGate Veridata Server.
2. Restart the Veridata Web Server.
3. Shutdown the Veridata Agent. Configure SSL for the Veridata Agent. See SSL Settings for Oracle GoldenGate Veridata Agent.
   a. Obtain the agent side keystores. See Using OPSS Keystore Service to Manage Veridata Keystores.
   b. Configure the agent-side keystores in the agent configuration properties file.
4. Run `configure_agent_ssl.sh` and supply the password to the keystores configured in the agent configuration file. See Modifying the Veridata Agent Wallet.
5. Start the agent.
6. If the trust is established properly between agent keystores and corresponding Veridata Server counterpart present in the OPSS Keystore Service, then SSL communication is established.

2.2.3 SSL Settings for Oracle GoldenGate Veridata Agent

By default, SSL is disabled for the Oracle GoldenGate Veridata Agent. To configure SSL, edit the following properties in the `agent.properties` file for your Veridata Agent.

Table 2–1 SSL Parameters in agent.properties file

Parameter	Description	Default Value
server.useSsl	Enables or disables SSL Communication between the Veridata Agent and Veridata Server. Possible values are: true: Enables SSL communication false: Disables SSL communication	false
server.use2WaySsl	Specifies whether the SSL communication is one-way or two-way. Options are: true: Uses two-way SSL communication false: Uses one-way SSL communication	false
server.identitystore.type	Specifies the type of keystore used for SSL configuration.	JKS
server.identitystore.path	Specifies the path for the server identity keystore.	./certs/serverIdentity.jks (Self Signed)
server.truststore.type	Specifies the type of truststore used for SSL configuration.	JKS
server.truststore.path	Specifies the path for the server truststore.	./certs/serverTrust.jks (Self Signed)
server.identitystore.keyfactory.alg.name	Algorithm name of the keyfactory used for SSL server identity store.	SunX509

Table 2–1 (Cont.) SSL Parameters in agent.properties file

Parameter	Description	Default Value
server.truststore.keyfactory.alg.nam	Algorithm name of the keyfactory used for SSL server trust store.	SunX509
server.ssl.algorithm.name	SSL algorithm name. **Note:** This value of this parameter must be same for the Veridata Agent and Veridata Server.	TLS

2.2.4 SSL Settings for GoldenGate Veridata Server

To enable SSL communication for all Veridata Server-Agent connections, you must set the SSL parameters in the `veridata.cfg` file located in the *`DOMAIN_HOME`*`/config/veridata` directory of your Veridata installation. Table 2–2 describes the various parameters that you must set in the `veridata.cfg` file for SSL communication.

You can also establish SSL communication only for certain connections. To do this, edit the connection properties in the Oracle GoldenGate Veridata web user interface. For more information, see the *Oracle GoldenGate Veridata Online Help*.

Note: In addition to these settings, configure Veridata Server Identity keystore and Trust keystore using the OPSS Keystore Service in the WebLogic domain. For more details, see Section 2.2.6, "Using OPSS Keystore Service to Manage Veridata Keystores".

Table 2–2 SSL Settings in veridata.cfg file

Parameter	Description	Default Value
server.useSsl	Enables or disables SSL Communication between the Veridata Agent and Veridata Server. Possible values are: true: Enables SSL communication false: Disables SSL communication	false
server.ssl.client.identitystore.keyfactory.alg.name	Algorithm name of the keyfactory used for SSL server identity store.	SunX509
server.ssl.client.truststore.keyfactory.alg.name	Algorithm name of the keyfactory used for SSL server trust store.	SunX509
server.ssl.algorithm.name	SSL algorithm name. **Note:** This value of this parameter must be same for the Veridata Agent and Veridata Server.	TLS

2.2.5 Creating Keystores and Self-Signed Certificates by using the Keytool Utility

For mutual authentication and to establish SSL communication, the Veridata Server and the Veridata Agents should mutually trust the add certificates in the respective truststores.

This section explains how to create keystores and self-signed certificates by using the `keytool` utility that is available as part of the Java Runtime Environment (JRE). For more details about `keytool`, refer Java documentation at `http://docs.oracle.com/javase/7/docs/technotes/tools/#security`.

2.2.5.1 Creating an Identity Keystore with a Self-Signed Certificate

The following `keytool` command creates a keystore containing a self-signed certificate:

```
keytool -genkey -keystore certs -keyalg rsa -alias vdt_alias -storepass server_ks_
pwd -keypass server_pwd
```

The `keytool` utility prompts to enter details about the certificate. Provide answers on the command-line when prompted.

2.2.5.2 Building Veridata Server and Veridata Agent Keystores

To build the Veridata Agent keystore, run the following `keytool` command:

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

To export the Veridata Agent certificate to a file, run the following `keytool` command:

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

To build the Veridata Web Server keystore, run the following `keytool` command:

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

To export the Veridata Server certificate to a file, run the following `keytool` command:

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

2.2.5.3 Importing Certificates to the Veridata Server and Agent Truststores

To import the Veridata Server certificate to the Agent truststore, run the following `keytool` command:

```
keytool -import -v -keystore agent.server.truststore -storepass ks_password -file
vdt.web.client.cer
```

To import the Veridata Agent certificate the Veridata Web Server truststore, run the following `keytool` command:

```
keytool -import -v -keystore vdt.web.client.truststore -storepass ks_password
-file agent.server.cer
```

2.2.6 Using OPSS Keystore Service to Manage Veridata Keystores

The Oracle Platform Security Services (OPSS) keystore services is used as a repository for storing identity and trust keystores for Veridata Server. The Veridata Agent keystores can also be managed using the OPSS keystore service. For more information, see "Managing Keys and Certificates with the Keystore Service" in *Securing Applications with Oracle Platform Security Services*.

Table 2–3 lists the default values for the OPSS settings.

Table 2–3 OPSS Settings

Setting	Value
Name of the Application Stripe created by WebLogic Server.	VeridataSec

Table 2–3 (Cont.) OPSS Settings

Setting	Value
Name of the identity keystore under the application stripe VeridataSec.	VeridataWebIdentityStore
Name of the trust keystore under the application stripe VeridataSec	VeridataWebTrustStore

To configure two-way SSL by using the OPSS keystore service:

1. For each Veridata Agent, create an identity and trust keystore.
2. Update the `VeridataWebIdentityStore` with the identity certificate of the Veridata Web Server.
3. Update the `VeridataWebTrustStore` with all Veridata Agent certificates.
4. Update each Veridata Agent truststore with the identity certificate of the Veridata Web Server.
5. Export the Agent keystore and truststore as JKS files and note the passwords.
6. Distribute the JKS files to corresponding Agent machines.
7. Run the `configure_agent_ssl` tool to update the Agent wallet with the keystore passwords.
8. For each Agent, configure the agent.properties file to enable SSL.

To configure one-way SSL by using the OPSS keystore service:

1. For each Veridata Agent, create an identity keystore.
2. Update the `VeridataWebIdentityStore` with the identity certificate of the Veridata Web Server.
3. Export the Agent keystore and truststore as JKS files and note the passwords.
4. Distribute the JKS files to corresponding Agent machines.
5. Run the `configure_agent_ssl` tool to update the Agent wallet with the keystore passwords.
6. For each Agent, configure the agent.properties file to enable SSL.

2.2.7 Modifying the Veridata Agent Wallet

Before you start the Veridata agent in SSL mode, you must update the Veridata Agent Wallet with the identity and trust keystore passwords. Otherwise, the Veridata Agent fails to start.

To update the wallet, run the `configure_agent_ssl` script that is available in the agent home:

AGENT_HOME`\configure_agent_ssl.sh` *AgentID*

where `AgentID` is the name of the agent properties file, without the `.properties` extension. The default value for `AgentID` is `agent`.

When prompted, enter the entry or unlock password for the identity and trust keystores for the agent.

2.3 Securing the Oracle GoldenGate Veridata Files

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

2.3.1 Controlling Access to the Installation Directories

Standard operating system permissions apply to the programs, files, and directories within the Oracle GoldenGate Veridata Server and Web User Interface, and Oracle GoldenGate Veridata Agent installation directories. You should adjust the permissions for these objects based on your business security rules.

2.3.2 Securing files that Contain Business Data

Oracle GoldenGate Veridata Server creates data files that will contain sensitive application data. By default, these files reside in the `DOMAIN_HOME/veridata/reports`. All of the sub-directories within that directory contain files that may reflect business data.

The types of files that contain sensitive data are:

- The comparison report (`rpt` sub-directory)
- The out-of-sync report (`oosxml` and `oos` sub-directories)

These files inherit the same file permissions as those of the user that runs the Oracle GoldenGate Veridata Server installation program. Do not change the permissions, or Oracle GoldenGate Veridata may be unable to maintain them. These files should be kept just as secure as you would keep your business data. Users of Oracle GoldenGate Veridata Web do not require access to these files because they see the same information through the client interface.

The contents of all report files can be optionally encrypted. For more information, see Section 2.5, "Encrypting Report Files".

2.4 Securing Access to Oracle GoldenGate Veridata by Defining User Roles

You can assign security roles to the users of Oracle GoldenGate Veridata to control their access to the functions that are performed by the software, some of which expose selected data values from the database. Table 2–4 describes the Veridata user roles.

Table 2–4 Veridata User Roles

Veridata	Type	Description
veridataAdministrator	Type-A	The administrator 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.
veridataPowerUser	Type-A	The power user 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.

Table 2–4 (Cont.) Veridata User Roles

Veridata	Type	Description
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, and view comparison reports and out-of-sync report information through the Oracle GoldenGate Veridata Web User Interface or at the file level.
veridataRepairOperator	Additional	The RepairOperator role can use the Repair feature in Veridata.
veridataCommandLineUser	Additional	The commandLineUser role provides access to the Veridata command line tools, vericom and veridata scripting.

These roles are categorized into various types 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 Veridata functions.

Security is controlled through the Oracle WebLogic Server Administration Console. From this interface, 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.

To open Oracle WebLogic Server Administration Console

1. Connect to the Oracle WebLogic Server Administration Console from a browser by typing the following address:

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

   Where:

   `weblogic_admin_server_hostname` is the name or IP address of the system where the Oracle GoldenGate Veridata server and web components are hosted, and *`admin_server_port`* is the port number assigned to the server (default is 7001).

2. Log on to the Administration Console as an Oracle GoldenGate Veridata administrator user. A default administrator user was created during the creation of Oracle GoldenGate Veridata domain.

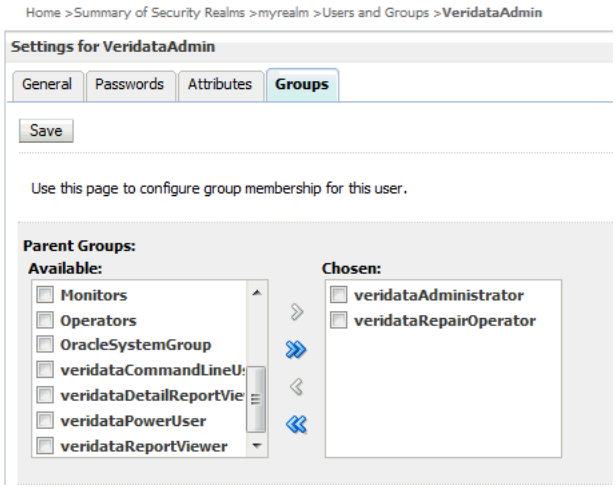
To create or edit a user

1. In the left pane of the Administration Console, select **Security Realms**.

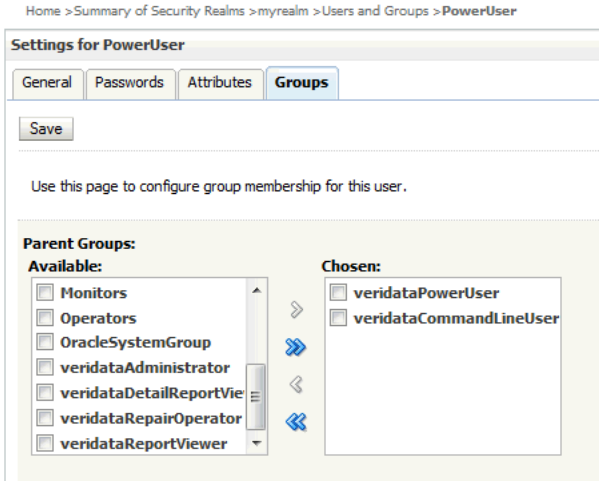
2. On the Summary of Security Realms page select the name of the Veridata security realm.
3. On the Settings for Veridata Security realm page select **Users and Groups** > **Users**. The User table displays the names of all users defined in the Authentication provider. Click New to create a new user or select an existing user to edit settings. Enter the following properties for a new user:
   - **Name**: Specify a name for the user.
   - **Provider**: Select the Authentication provider for the user.
   - **Password**: Enter a password for the user.
   - **Description**: (Optional) Specify a description for this user.
4. To assign a role to the user, go to the Settings for *user_name* page and click Groups.

   Select appropriate roles for the user. All roles for a Veridata user are described in Table 2–4, " Veridata User Roles".

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



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



5. Click **Save** to save the user.

2.5 Encrypting Report Files

Oracle GoldenGate Veridata provides you with an option to encrypt the comparison report files (.rpt, .oos, .oosxml). The following sections explain the report encryption in Veridata:

- Section 2.5.1, "Enabling Report Encryption"
- Section 2.5.2, "Using the reportutil Tool to view Reports"

2.5.1 Enabling Report Encryption

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

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

To enable encryption of Veridata report files, set `server.encryption` to `true`.

Encryption of Veridata report files use 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 `server.encryption.bits` parameter in `veridata.cfg`. Note that any encryption strength greater than 128 requires you to use a JRE that has Unlimited Strength Cryptography Extension installed.

For more information about these parameters, see "Parameters for Report File Encryption".

Encrypted Veridata 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 XML chunk file (`NNN` is a decimal number)

2.5.2 Using the reportutil Tool to view Reports

When Veridata report encryption is enabled, all Veridata report files will be encrypted using an encryption key which is initially a large random value. The encryption key can be changed if required.

The encrypted files must be decrypted before you read it. The Veridata Web User Interface automatically decrypts files before displaying them. Alternatively, use the `reportutil.sh/.bat` utility to display the encrypted contents. This utility is located in the *`VERIDATA_DOMAIN_HOME`*`\veridata\bin` directory. Run the tool as follows:

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

Where `wlport` is the WebLogic Server port number (the default port is 8830) and `wluser` represents the WebLogic 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. Or else the command prints the decrypted file to the standard output.

Note that the Veridata user running the `reportutil` tool must be in the appropriate user group to perform the operations:

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

For more information about the user roles, see Section 2.4, "Securing Access to Oracle GoldenGate Veridata by Defining User Roles".

3

Running the Oracle GoldenGate Veridata Programs

This chapter explains how to run the Oracle GoldenGate Veridata programs, such as the agents and Java components.

This chapter includes the following sections:

3.1 Starting and Stopping the C-agent and Manager

The C-agent starts automatically at the request of Oracle GoldenGate Veridata Server when initiating comparisons. However, for Oracle GoldenGate Veridata 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 an automatic process, you can stop the Manager process that controls the agent. Stopping Manager prevents Oracle GoldenGate Veridata 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 Oracle GoldenGate Veridata Agent installation location, run GGSCI.
2. In GGSCI, issue the appropriate command as follows to stop or start the Manager.

```
START MANAGER
```

Or...

```
STOP MANAGER
```

To control the C-agent Manager as a Windows service

If Manager is installed as a Windows service, it either starts at system boot time or must be started manually, depending on how the service is configured. The installation default is to start automatically at system boot time. To start the service

manually, use the START MANAGER command in GGSCI or start the service in the **Services** control panel applet.

To change the startup behavior of the service, right click the name in the **Services** control panel, and then select **Properties**.

When Manager is installed as a Windows service, you can stop it in the **Services** control panel applet or with the STOP MANAGER command in GGSCI.

3.2 Starting and Stopping the Java-Based Components

The Oracle GoldenGate Veridata Server and Oracle GoldenGate Veridata Web components are Java-based programs. The Oracle GoldenGate Veridata 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.

3.2.1 Controlling the Java-Based Components from the Command Line

To control the agent component, change directories to its installation directory and issue the appropriate command as follows:

UNIX and Linux	Windows
`agent.sh {start | run}`	`agent.bat {start | run}`
Or...	Or...
`agent.sh stop`	`agent.bat stop`

Where:

- `run` starts the agent in the same command window from which 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 `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 Log Information

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

Use the following command:

UNIX and Linux	Windows
`agent.sh reloadLog`	`agent.bat reloadLog`

3.4 Connecting to Oracle GoldenGate Veridata Web Interface

To connect to the Veridata Web User Interface, open a web browser and type the following address:

`http://`*`hostname`*`:`*`port`*`/veridata`

Where:

`hostname` is the name of the system where Oracle GoldenGate Veridata Server is installed and *`port`* is the port number where it is running (default is 8830). Use `localhost` as the host name if connecting on the system that is local to the server installation.

Examples:

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

The Oracle GoldenGate Veridata Web login page is displayed upon successful connection. Log in with your user name and password. For full instructions on using Oracle GoldenGate Veridata Web Interface, see the Online Help.

4

Running Comparisons from the Command Line

This chapter explains how to use the `vericom` command line interface to run comparisons.

This chapter includes the following sections:

4.1 Overview of the Command Line Interface

You can use the `vericom` tool of Oracle GoldenGate Veridata 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 interface

Comparisons also can be run from the Oracle GoldenGate Veridata Web interface. This interface provides greater control in configuring the objects to be compared and for controlling runtime parameter settings.

4.2 Running Vericom

The vericom program can be run by anyone who has the correct operating system permissions to run it.

1. On the system where the Oracle GoldenGate Veridata is installed, run the command shell of the operating system.
2. Navigate to the *VERIDATA_DOMAIN_HOME*/veridata/bin directory.
3. Use the following syntax to run the vericom program.

Syntax

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

Required Parameters

One of the following are required; otherwise an error is returned. Enter only one option.

```
[-wlport port ] |
-wluser user_name |
-help |
-helprun |
[-version | -v] |
[-job | -j] job |
```

The -wluser specifies the WebLogic Server user name to connect to the WebLogic Server. This WebLogic 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 Veridata Scripting tool.

See Section 2.4, "Securing Access to Oracle GoldenGate Veridata by Defining User Roles".

If -version, -v, -help, or -helprun are specified, they take precedence over any other flag specified.

Optional Parameters

These are the optional parameters:

```
[ -g group -c compare_pair ]
[ -nw ]
[ -rP profile ]
[ -rR ]
| -rO ]
[ -rN threads ]
[ -rD seconds ]
[ -rC | +rC ]
[ -rOb | -rOx | -rO2 | -rO0 ]
[ -rOs records ]
[ -rTi ]
[ -rTc ]
[ -rTs trace_number ]
[ -pS source_partition_name |
    -pSq source_sql_predicate |
    -pSA1 source_ascii_start_key |
    -pSA2 source_ascii_end_key |
    -pSH1 source_hex_start_key |
    -pSH2 source_hex_end_key ]
```

```
[ -pT target_partition_name |
    -pTq target_sql_predicate |
    -pTA1 target_ascii_start_key |
    -pTA2 target_ascii_end_key |
    -pTH1 target_hex_start_key |
    -pTH2 target_hex_end_key ]
[ -pq sql_predicate ]
[ -rd0 | -rdN run_ID ]
[ -wp ]
```

Table 4–1 Vericom Runtime Arguments

Argument	Description
`-wluser`	Specifies the WebLogic Server user name that authenticates and connects to the server.
`-wlport`	Specifies the WebLogic Server port number.
`-help`	Displays the `vericom` syntax components and their descriptions.
`-helprun`	Displays run-related syntax components and their descriptions.
`{-version | -v}`	Displays the version of the Oracle GoldenGate Veridata command-line interface that is being used.
`{-job | -j}` *job*	Specifies the job to be run. For *job*, specify the name that was assigned when the job was created in Oracle GoldenGate Veridata Web.
`-g` *group* `-c` *compare_pair*	Specifies a group and compare pair. For *group* and *compare_pair*, specify the names that were assigned when these objects were created in Oracle GoldenGate Veridata Web. ■ If `-g` and `-c` are used, `-j` must also be used.
`-nw`	Directs `vericom` not to wait for the job to finish before returning the prompt. Instead, `vericom` returns immediately after starting a job.
`-rP` *profile*	Overrides the profile that is defined for a job. For *profile*, specify the name that was assigned when the profile was created in Oracle GoldenGate Veridata Web. ■ If `-rP` is used, -j must be used.
`-rR`	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. ■ Do not use `-rR` and `-r0` in the same run.
`-r0`	A run override option. Generates an `OOSXML` file 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. ■ `-r0` must be used with `-j`. ■ Do not use `-rR` and `-r0` in the same run.

Table 4–1 (Cont.) Vericom Runtime Arguments

Argument	Description
`-rN` *threads*	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 `-j` or if just one comparison is run `by` using `-j` with `-g` and `-c`.
`-rD` *seconds*	Delays the confirmation step by the specified number of seconds to account for replication lag. Delaying the confirmation step reduces the number of false out-of-sync results that occur because an updated source value was not replicated fast enough. This option overrides the default job profile and has no effect if the `-rR` option is used.
`-rC` | `+rC`	Controls whether or not the confirmation step (confirm OOS) is performed in the job. ■ `-rC` skips the confirmation step. You can skip the confirmation step if activity on the source tables is stopped or if replication is not continuously updating the target table(s). ■ `+rC` includes the confirmation step. These options override the default job profile and are mutually exclusive. They have no effect unless `-j` is used.
`-rOb` | `-rOx` | `-rO2` | `-rO0`	Controls the kind of file that is produced for the out-of-sync report. ■ `-rOb` generates binary format that is compatible with the Oracle GoldenGate Veridata Web browser. ■ `-rOx` generates output in XML. ■ `-rO2` generates both binary and XML output. ■ `-rO0` suppresses out-of-sync output. These options override the default job profile and are mutually exclusive. They have no effect if `-rR` is used.
`-rOs` *records*	Limits the number of out-of-sync rows that are written to a chunk of the `OOSXML` 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 `-rR` is used.
`-rTi`	Turns on tracing of Oracle GoldenGate Veridata Agent for the initial comparison step. Do not use without the guidance of an Oracle support analyst.
`-rTc`	Turns on tracing of Oracle GoldenGate Veridata Agent for the confirmation step. Do not use without the guidance of an Oracle support analyst.
`-rTs` *trace_number*	Turns on tracing for Oracle GoldenGate Veridata Server. *trace_number* is a bitmask of server execution trace flags. The higher the level, the more detailed the trace data. Do not use without the guidance of an Oracle support analyst.

Table 4–1 (Cont.) Vericom Runtime Arguments

Argument	Description
-pS *source_partition_name* | -pSq *source_sql_predicate* | -pSA1 *source_ascii_start_key* | -pSA2 *source_ascii_end_key* | -pSH1 *source_hex_start_key* | -pSH2 *source_hex_end_key*	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. **-pS *source_partition_name*** 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 Veridata Server. There will be an error if the specified partition does not exist. **-pSq *source_sql_predicate*** 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. 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. If the predicate contains special characters (such as $, *, < in sh/csh or %, < in Windows), they must be properly escaped for that shell or interpreter. **-pSA1 *source_ascii_start_key*** Specifies an ASCII key as the starting key value of a partition that overrides an existing source partition for an Enscribe file. **-pSA2 *source_ascii_end_key*** Specifies an ASCII key as the ending key value of a partition that overrides an existing source partition for an Enscribe file. **-pSH1 *source_hex_start_key*** Specifies a hexadecimal key as the starting key value of a partition that overrides an existing source partition for an Enscribe file. **-pSH2 *source_hex_end_key*** Specifies a hexadecimal key as the ending key value of a partition that overrides an existing source partition for an Enscribe file.
-pT *target_partition_name*| -pTq *target_sql_predicate* | -pTA1 *target_ascii_start_key* | -pTA2 *target_ascii_end_key* | -pTH1 *target_hex_start_key* | -pTH2 *target_hex_end_key*	These options specify target partitions and have the same rules as the corresponding options that specify source partitions.
-pq *sql_predicate*	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 -pSq *source_sql_predicate* and-pTq *target_sql_predicate*.

Table 4–1 (Cont.) Vericom Runtime Arguments

Argument	Description
`-rd0 | -rdN run_ID`	Valid for NonStop SQL/MP and Enscribe only. Controls delta processing for a compare pair. ▪ `-rd0` disables delta processing for this run. All rows are compared. ▪ `-rdN run_ID` enables delta processing using a previous job run as the basis for the delta. For *run_ID* 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: ▪ `-j` ▪ `-g` ▪ `-c`
`-wp seconds`	Waits for a job to complete. The client also polls the status of job submitted to the server at the specified interval (in seconds). ▪ `-wp` must be used with `-job | -j`.

4.3 Vericom exit statuses

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

Status	Description
0	The command executed successfully. If a job was run, it finished with all rows in-sync. If `-nw` was specified, the exit status is 0 if the job started successfully.
1	Invalid `vericom` syntax was used. For example, the following are invalid: `vericom.sh -helptun` (Typographical error.) `vericom.sh -j -g group1` (The name of the job is missing.)

3	Provides more granularity for input errors that involve flags that run comparisons. For example, the following mistakes will cause this error:
	`vericom.sh -j job1 -c address=address`
	In the preceding example, the `-g group` input is missing. It is required with `-j` if `-c` is used.
	`vericom.sh -j job1 -g group1 -rd0`
	In the preceding example, the `-rd0` flag requires `-c` because delta processing applies at the compare pair level.

4	The job ran successfully, but there were rows that had a comparison status of something other than in-sync.
5	There was a communication error with Oracle GoldenGate Veridata Server.

4.4 Vericom Output Examples

To view the results of a comparison that is run with `vericom`, you can use Oracle GoldenGate Veridata Web to view the comparison report (see Section 1.4, "Viewing Comparison Results"), and you can view the output that is returned by `vericom` to the terminal. If a run finishes successfully, statistics for the job are displayed.

Example 1

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

```
VERIDATA_DOMAIN_HOME\veridata\bin\vericom.bat -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

The following example shows a run of `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 -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

The following example shows a run of compare pair `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 -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 of the previously executed command is in the special variable `'$?'` if you are using `SH` or `KSH` shells. If you are using the `CSH` shell, the exit status of the previously executed command is in the special variable `'$status'`.

5

Configuration Scripting

In addition to using the Oracle GoldenGate Veridata Web User Interface, you can use scripting to define portions of your configuration. This chapter includes the following topics:

- Introduction to Scripting
- Elements

5.1 Introduction to Scripting

With scripting 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 `web/bin` directory of the Oracle GoldenGate Veridata installation location. You can refer to the online help for detail on the input since the DTD entries mostly mirror the Oracle GoldenGate Veridata Web User Interface.

This documentation assumes that the user has a knowledge of basic XML and its rules.

This scripting has the following advantages:

- It can reduce the time needed 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 scripting supports configuring:

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

Oracle GoldenGate Veridata scripting does *not* support configuring:

- Profiles
- Overriding data type formats on connections

5.1.2 Running the Scripting Utility

The scripting utility runs from the *DOMAIN_HOME*`/veridata/bin` directory of the Oracle GoldenGate Veridata installation location. The Windows version is named `veridata_scripting.bat` and the UNIX and Linux is `veridata_scripting.sh`.

The syntax for running the script on Windows is:

```
veridata_scripting.bat [-create | -update | -delete] configuration.xml
```

- 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.
- *`configuration.xml`:* The name of scripting XML configuration file that you created to describe the configuration. This is a required entry.

The scripting utility returns an exit status of 0 if successful and non-zero if an error was encountered. More information on the errors can be found in the `shared logs/scripting.log` file in the installation location.

5.1.3 Processing the Configuration

The scripting 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 `shared logs/scripting.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 hypen (-) 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.

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 Elements

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.

CHANGE THE ELEMENTS TO BRIDGEHEAD?

5.2.1 configuration

The main element is `configuration`.

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

Table 5–1 configuration Elements

Elements	Description
`connection`	One or more Oracle GoldenGate Veridata Web User Interface database connection definitions.
`group`	One or more comparison group definitions.
`job`	One or more job definitions.

The following attributes describe the `configuration` element:

Table 5–2 configuration Attributes

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

Example

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

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

5.2.2 column

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

The following attributes describe the column element:

Table 5–3

Attribute	Description
source-name	A regular expression that defines a set of source column names. This value is *required*.
target-name	A regular expression that defines a set of target column names. It can include references to groups captured by the source-name expression.
exclude	Indicates whether or not the matched columns should be excluded from the compare pair. The options are: "true" - The matched columns should be excluded. "false" - The matched columns should be included. This is the *default*.

Table 5–3 (Cont.)

Attribute	Description
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. QUESTION: okay? The values can be any of the data types supported by Oracle GoldenGate Veridata.
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 timezone to override the default timezone of the comparison.

5.2.3 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 5–4 compare-pair Elements

Element	Description
enscribe-info	One or more sets of information used when comparing NonStop Enscribe files.
partition	One or more specifications of a subset of tables.
column	One or more definitions of a set of columns to be included or excluded.

The following attributes describe the compare-pair element:

Table 5–5 compare-pair Attributes

Attribute	Element
name	An expression defining the name of the compare pair. This expression can include groups captured with source-table expressions and target table group $0.
source-table	A regular expression that defines the table or tables to be compared. See "Regular Expression Grouping" later in this section for more detail. The default is to match all tables.
target-table	A regular expression that defines the target tables for the comparison. This may contain references to groups captured by the source table expression. The default is $0 for the full source table name.

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

Attribute	Element
source-schema	The name of the default schema for the source tables referenced for the compare pair. QUESTION: SHOULD THE FOLLOWING BE INCLUDED FOR THE GROUP DEF OF SOURCE-SCHEMA? The default is the value specified for the group. For SQL/MP, this is the subvolume of the SQL catalog. This is not used with Enscribe files.
target-schema	The name of the default schema for the target tables referenced for the compare pair. The default is the value specified for the group. For SQL/MP, this is the subvolume of the SQL catalog. This is not used with Enscribe files.
source-catalog	The default catalog for the source tables referenced in this compare pair. QUESTION (AS ABOVE) For SQL/MP, this is the volume of the SQL catalog. This is not used for Oracle, DB2, Enscribe, or Teradata.
target-catalog	TThe 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 Oracle, DB2, Enscribe, or Teradata.
exclude	Indicates whether or not the compare pair should be included in the group element. This can be used to remove a compare pair generated by an earlier compare pair element. The options are: "true" - Exclude the compare pair. "false" - Include the compare pair. This is the *default*.
source-file-pattern	The default file pattern for the source if the data source is Enscribe or SQL/MP.
target-file-pattern	The default file pattern for the target if the data target is Enscribe or SQL/MP.
source-pkey	The name of the unique index to use as the source portion of the user-specified primary key. The default is no user-specified index name.
target-pkey	The name of the unique index to use as the target portion of the user-specified primary key. The default is the value of the source-pkey.
delta-processing	Indicates whether or not delta processing is enabled for this compare pair. The options are: "true" - delta processing is enabled. "false" - delta processing is not enabled. This is the *default*.
profile-name	The name of the profile to use when running the compare-pair comparison.
system-key	If the compare pair has no column elements and no specified source-pkey, Oracle GoldenGate Veridata will select the most appropriate primary key or unique index to use. The options are: "true" - Oracle GoldenGate Veridata selects the key if it is not defined. This is the *default*. "false" - Oracle GoldenGate Veridata does not select the key.
system-columns	Indicates that the compare pair contains column elements with the type attribute set to key, so the generated compare pair will have user-defined columns for the key. The options are: "true" - Compare pair has key column elements. This is the default. "false" - Compare pair does not have key column elements.

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

Attribute	Element
wildcard	Specifies the pattern matching method that is used. The options are: "ggs" - Use the typical Oracle GoldenGate pattern that matches an asterisk (*) to any number of characters. "regex" - Use regular expressions for matching. "default" - Use the setting for the configuration. This is the *default*.

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. 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. (QUESTION: why variable portion captured here and initial letter for the third bullet point?)
- [^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.

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

Example

QUESTION: HOW TO DEFINE THIS EXAMPLE? SHOULD IT BE AN EXAMPLE THAT IS NOT IN THE SAMPLE FILE? The following example describes the key-only compare-pair. Its source tables are defined in the test schema and target tables in the other schema. It will exclude pairs where the source column name begins with S and the target column name begins with T and otherwuse matches the source column name.

```
.
.
.
<compare-pair name="key-only" source-schema=test target-schema=other
   source-table="S(.*)" target-table=T$1">
   <column source-name="(.*)" target-name="$1" exclude="true"/>
</compare-pair>
.
.
.
```

5.2.4 connection

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

The following element can be nested within the connection element :

Table 5–6 connection Elements

Element	Description
`description`	Provides a description of the connection.

The following attributes describe the `connection` element:

Table 5–7 connection Attributes

Attribute	Description
`name`	A name that identifies the connection. This is a *required* entry.
`host`	The name of the system on which the Oracle GoldenGate Veridata agent is running.
`port`	The port number of the system on which the agent is running.
`user`	The user name the agent uses to connect to the database.
`password`	The password the agent uses to connect ot the database.
`agent-timeout`	The amount of time Oracle GoldenGate Veridata will wait before timing out when sending requests to the agent. WE MAY DELETE THIS ONE. QUESTION: *Is this correct? what time interval, seconds?*
`truncate-spaces`	Either `"true"` or `"false"` to indicate whether or not spaces will be removed from the end of character columns. The default is `"true"` to truncate spaces.
`fetch-size`	(Oracle only) The number of rows fetched in each batch.

Example

The following example identifies the connection named `source`.

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

5.2.5 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>
```

5.2.6 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 5–8 enscribe-info Elements

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

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

Table 5–9 enscribe-info Attributes

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

5.2.7 enscribe-key

The `enscribe-key` element defines the key that is to be used for Enscribe files. The `enscribe-key` element has no nested elements or text data.

The following attributes describe the `enscribe-key`:

Table 5–10 enscribe-key Attributes

Attribute	Description
`start-key`	The key that is to be used to begin reading the Enscribe file. This is a *required* entry.
`end-key`	The key of the last Enscribe record that should be read. This is a *required* entry.
`format`	Specifies the format of the Enscribe key. The options are: `"ascii"` - The format of the key is ASCII. This is the *default*. `"hexadecimal"` - The format of the key is hexadecimal.

5.2.8 expandddl

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

The following attributes describe the `expandddl` element:

Table 5–11 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.
ddl-separator	The character separator for defining array output into columns. An example is the dash used in FIELDX-3, which is the third occurance of FIELDX in the array. The options are: "none" - There is no separator. This is the *default*. "dash" - Use a dash (-) as the separator. "bracket" - Use brackets [] as the separator. "underscore" - Use underscore (_) as the separator. "double-underscore" - Use double underscore (__) as the separator.
zero-fill-length	Prepends zeros to adjust the number of the occurance. 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 max-col-name-length. The options are: "true" - Fix the names that exceed the maximum. This is the *default*. "false" - Do not change the names that exceed the maximum.
max-col-name-length	The maximum length allowed for a column name. The entry is a number within quotation marks. The *default* is "120".

5.2.9 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 5–12 Group Elements

Element	Description
description	Provides a description of the group.
partition	Defines an Oracle GoldenGate Veridata partition that will be applied to all the compare pairs in the group. The parition specifies subsets of the data.

Table 5–12 (Cont.) Group Elements

Element	Description
compare-pair	Defines one or more compare pairs. The compare-pair elements are added to the group in the order they are specified. If the same compare pair fits the criteria of another specification in the group, the first compare pair will be used.

The following attributes describe the group element:

Table 5–13 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 repostiory. "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 validataion. 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

QUESTION: WHAT SHOULD WE DEFINE FOR THE GROUP?

```
<group name="weekly-tables" source-conn="source" target-conn"="target">
   <description>
   .
```

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

5.2.10 job

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

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

Table 5–14 job Elements

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

The following attributes describe the `job` element:

Table 5–15 job Attributes

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

Example

QUESTION:

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

5.2.11 partition

The `partition` element defines a subset of the tables that will be compared.

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

Table 5–16 partition Elements

Element	Description
`query`	A Boolean SQL expression used in the where clause applied to the initial comparison query for the source and target.
`source-query`	A Boolean SQL expression used in the where clause applied to the initial comparison query for the source.
`target-query`	A Boolean SQL expression used in the where clause applied to the initial comparison query for the target.
`enscribe-key`	An Enscribe key value to be applied to the source and target files when comparing Enscribe files.
`source-enscribe-key`	An Enscribe key value to be applied to the source file when comparing Enscribe files.
`target-enscribe-key`	An Enscribe key value to be applied to the target file when comparing Enscribe files.

The following attributes describe the `partition` element:

Table 5–17 partition Attributes

Attribute	Description
`name`	A name that identifies the partition. This is a *required* attribute.
`type`	Indicates the type of selection to be used. The options are: `"sql"` - Specifies a SQL expression.This is the default. `"enscribe"` - Specifies an Enscribe key specification.
`default`	Indicates whether this is the default partition that will be automatically applied to the compare pair at run time. It corresponds to the "use at run time" option in the Oracle GoldenGate Veridata Web User Interface. The options are: "true" - This is the default partition. "false" - This is not the default partition. This is the *default* value.

Example

QUESTONS:WHAT DOES THIS DO? DO WE WANT A DIFFERENT EXAMPLE?

```
<partition name="replicate" type="sql" default="true">
  <source-query>
    &quot;replicated&quot; =apos;false&apos;
  </source-query>
  <target-query>
    &quot;replicated&quot; =apos;true&apos;
  </target-query>
</partition>
```

5.2.12 query, source-query, target-query

The `query`, `source-query`, and `target-query` elements are free-form text that can be used to attach a query to the containing element. All three define Boolean SQL expressions that can be used in a where clause of the initial comparison query.

The `query` elements have no attributes. The `source-query` element is applied to the source, the `target-query` to the target, and `query` to both.

6

Oracle GoldenGate Veridata Server Configuration Parameters

This chapter describes parameters that adjust different aspects of the sort memory configuration when using server-side sorting.

This chapter includes the following sections:

6.1 Overview of the Server Memory

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

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

- **IPC buffer memory**. This is the memory that is used to exchange messages between the server and the agent.
- **Scratch runtime transient memory**. This is virtual memory space.

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

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

6.2 Estimating Memory Usage

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

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

To Estimate Memory based on the Number of Concurrent Comparisons

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

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

`server.max_comparison_sort_memory` * `server_max_comparison_threads`

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

To Estimate the Amount of Memory Used per Row

Refer the section "Disk and Memory Requirements for the Server Component" in *Installing and Configuring Oracle GoldenGate Veridata* for the calculation to estimate the amount of memory used per row.

6.3 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. For more information, go to `http://support.oracle.com`.

6.4 Parameter Descriptions

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

Server Parameters

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

- server.veridata_data
- server.persistence_db_type
- server.meta_session_handle_timeout
- server.max_concurrent_jobs
- server.max_concurrent_comparison_threads
- server.max_sort_memory
- server.max_comparison_sort_memory
- server.sort_waiting_threshold
- server.concurrent.writers
- server.concurrent.readers
- server.number_sort_threads

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.

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
- MYSQL

Default Value

ORACLE_OCI

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

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

server.max_concurrent_comparison_threads

Sets the maximum number of concurrent comparisons that can be executed. In general, the amount configured by the server is the optimal value, given the machines resources. You can lower this number to reduce the impact of the server on your system. When this limit is reached, no new comparisons will start until an active comparison completes.

Syntax

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

- `default` allows Oracle GoldenGate Veridata to compute the maximum number of concurrent threads based on the `server.max_comparison_sort_memory` and available resources. The default value is the server.max_sort_memory divided by `server.max_comparison_sort_memory`.
- *number* is a positive integer that sets the maximum number of concurrent comparison threads.

Example

```
server.max_concurrent_comparison_threads 100
```

Default Value

The default value is based on the available virtual memory.

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 default amount is the Java boot option `-Xmx` maximum memory setting less the 200 MB needed for basic tasks. You can limit this amount to make more memory available for the Oracle GoldenGate Web User Interface.

If a comparison cannot get enough virtual memory when it starts, it fails or tries again depending on the setting of the `server.sort_waiting_threshold` parameter. If a comparison does get enough virtual memory, the currently available sort virtual memory gets decremented by the amount that the comparison reserves. When a comparison completes, it increments the amount of available sort virtual memory by the amount of sort virtual memory that it had reserved.

Syntax

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

- `default` allows Oracle GoldenGate Veridata to define a maximum value that is dependent on the operating system.
- *number*`{M | m}` specifies a value in megabytes. For example, 1000M means a limit of 1000 megabytes. If this number exceeds the amount of available memory, the value will be reduced to the amount of available memory.

Example

```
server.max_sort_memory 1000M
```

Default Value

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

server.max_comparison_sort_memory

Sets the maximum amount of virtual memory that can be reserved by a single comparison that is using server sorting. A comparison that tries to reserve more than this amount will fail, even if it is configured to retry.

The amount set for this property limits the amount that can be set in the Oracle GoldenGate Web User Interface Profile.

If the value of this parameter is changed, it might cause the concurrent comparisons that use server-side sorting to fail or retry, depending on the sorting parameters defined in the profile that is being used.

Syntax

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

- `default` is the value of `server.max_sort_memory`.
- *number*`{M | m}` specifies a value in megabytes. For example, 600M sets the limit for any comparison to be 600 megabytes.

Example

```
server.max_comparison_sort_memory 1000M
```

Default Value

The default value is the value of `server.max_sort_memory`.

server.sort_waiting_threshold

Sets a threshold above which new jobs will fail if there are too many comparisons waiting to retry a virtual memory reservation.

Syntax

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

- `default` is zero; if memory cannot be allocated the job will not be allowed to run.
- *`number`* sets the number of comparisons that can be waiting before new jobs will fail. For example, a value of 100 means that if there are 100 comparisons waiting to retry a memory reservation, and there is a job that is ready to begin execution, this job will not be allowed to run.

Example

```
server.sort_waiting_threshold 1000M
```

Default Value

The system calculates the value.

server.concurrent.writers

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

Syntax

```
server.concurent.writers number
```

Example

```
server.concurent.writers number
```

Default Value

The default value is 4.

server.concurrent.readers

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

Syntax

```
server.concurent.readers number
```

Example

```
server.concurent.readers number
```

Default Value

The default value is 4.

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 default value is 2.

Parameters for Configuring SSL Communication

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

- server.useSsl
- server.ssl.client.allowTrustedExpiredCertificates
- server.ssl.client.identitystore.keyfactory.alg.name
- server.ssl.client.truststore.keyfactory.alg.name
- server.ssl.algorithm.name

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.

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.

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.

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.

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.

Parameters for Veridata Command-Line Utility

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

- veridata.cli.run_from_managed_server
- veridata.cli.managed_server_name
- veridata.cli.server.listenAddress
- veridata.cli.server.timeout.seconds

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.

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`.

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`.

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.

Parameters for Report File Encryption

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

- server.encryption
- server.encryption.bits

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.

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.

A

Moving from a Test to Production Environment

This appendix describes how to move an Oracle GoldenGate Veridata installation from a test environment (a source environment) to a production environment (a target environment).

This appendix includes the following sections:

- Moving Installations from a Source Environment to a Target Environment
- Additional Steps for Moving Oracle GoldenGate Veridata Repository

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.

For more information about using these scripts, see "Common Procedures for Moving to a Target Environment" in *Administering Oracle Fusion Middleware*.

A.2 Additional Steps for Moving Oracle GoldenGate Veridata Repository

In addition to the common procedures described in the guide *Administering Oracle Fusion Middleware*, 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 `ORACLE_HOME/veridata/t2p/scripts` directory.

Note: You must have the tools sqlplus, impdp, and expdb installed on the system where you run the scripts.

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

```
ORACLE_HOME/veridata/t2p/scripts/oracle_exp_imp.sh  --export --dba_user system
--db_inst
ora11203 --vdt_user DEV31_VERIDATA --work_dir /tmp/work1
```

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

```
ORACLE_HOME/veridata/t2p/scripts/oracle_exp_imp.sh  --import --dba_user system
--db_inst ora11203
--vdt_user DEV32_VERIDATA --work_dir /tmp/work1
```

Table A–1 describes the arguments you specify while running the `oracle_exp_imp.sh/oracle_exp_imp.bat` script.

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.
dba_user <user_name>	Specifies a database user with DBA privileges.
db_inst <oracle_sid>	Specifies the database connect descriptor name, basically defined in the `$TNS_ADMIN/tnsnames.ora` file.
vdt_user <veridata_user>	Specifies the Veridata user or schema name for which the export or import is being done.
work_dir <work_dir>	Specifies the temporary directory for database dump and to save logs.
pwd_file <dba password file>	Specifies the file that contains the DBA user password.

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, execute the following statements as appropriate to your database:

For all databases:

```
UPDATE 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
```

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
```

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.

The statements for various database repositories are given below.

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%'
```

For MYSQL:

```
Update COMPARE_PAIRS SET  SRC_TABLE_NAME = CONCAT( '\SPROD.$PDATA' ,
    SUBSTRING(SRC_TABLE_NAME, 12)) Where SRC_TABLE_NAME like '\STEST.TDATA%'

Update COMPARE_PAIRS SET  TARG_TABLE_NAME = CONCAT('\TPROD.$PDATA',
    SUBSTRING(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. Start the Veridata agent after making these changes.

For example, create an xml as below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration SYSTEM
"http://@VeridataServerHost@:@veridataServerPort@/veridata/configuration.dtd">
    <configuration validation="required">
    <connection name="@conneciotnName@" 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>

VERIDATA_DOMAIN_HOME/veridata/bin/veridata_scripting.sh -update
/tmp/con.xml   -wlUser <cmd user>
```

B

Sample Configuration File

This appendix provides the contents of the following sample configuration file for using with the Oracle GoldenGate Veridata Scripting utility.

For more information about the parameters used in this configuration file, see Section 5.2, "Elements".

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 Section 5.2, "Elements".

```
<?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="server.us.company.com" port="9999"/>
    <connection name="nskScriptingTarget" host="server.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>
        <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.$FSS02.FSSVSRC.*"
source-table="TELLER"  target-file-pattern="\ZEUS.$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.$FSS02.FSSVSRC.*"
source-table="TELLER"  target-file-pattern="\ZEUS.$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="&quot;abc 11&quot;" target-schema="&quot;abc 11&quot;">
         <description>
            <![CDATA[
                SQL group with simple quoted schema and quoted table name
            ]]>
          </description>
        <compare-pair source-table="&quot;Quoted Table&quot;" target-table="*"  />
        <compare-pair source-table="&quot;Quoted*Table&quot;" 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>
```

Glossary

agent

See Oracle GoldenGate Veridata Agent.

client

An interface to Oracle GoldenGate Veridata Server, from which you configure or execute comparison work. Oracle GoldenGate Veridata provides a GUI client named Oracle GoldenGate Veridata Web and a command client named `vericom`, which provides a subset of the functions that can be performed from the web interface.

column mapping

An Oracle GoldenGate Veridata Web configuration object that establishes the correct structural relationship between the source and target objects in a compare pair.

compare group

See group.

compare pair

A configuration object in Oracle GoldenGate Veridata Web that associates one source table or file with one target table or file for the purpose of comparing their data. The source table typically contains data that is known or presumed to be up-to-date and accurate. The target table is a different table which contains identical data, such as a backup, that is synchronized with that of the source by some mechanism, such as replication.

comparison format

A standardized format to which Oracle GoldenGate Veridata transforms the native data type of a column. The use of standardized formats enables Oracle GoldenGate Veridata to support heterogeneous comparisons (those between different types of databases) where source and target data types are different, but similar.

comparison report

A report that is created at the conclusion of a comparison that contains information about the objects that were compared, the number of rows that are out-of-sync, and statistics about performance.

confirmation step or confirm-out-of-sync (COOS) step

A follow-up comparison that is performed after the initial comparison step to confirm a row's status as in-sync or perpetually out of synchronization. The confirmation step waits until after a specified replication latency threshold has expired, to give a replication mechanism time to post any change that was in-flight.

connection

A configuration object in Oracle GoldenGate Veridata Web that stores information about how the Oracle GoldenGate Veridata Server process will connect to a Veridata Agent process. It consists of a DNS host name or IP address, a Manager port number, and a data source type.

data source

A set of stored data, such as a relational or hierarchical database.

delta processing

A performance feature that Oracle GoldenGate Veridata supports on the NonStop platform by which Oracle GoldenGate Veridata detects which data blocks in the database were modified since a previous comparison and only compares the rows in those blocks. Rows in unchanged blocks are skipped.

digital signature

A message signed with a private key that can be verified by a person or process that has the public key, thereby proving that the message is unchanged and authentic.

GGSCI

The command line interface to the Oracle GoldenGate replication software, which also is used as the command interface to the Manager process of the Oracle GoldenGate Veridata Agent.

GLOBALS file

A parameter file that is required on some platforms where Oracle GoldenGate Veridata Agent is installed. It stores parameters that may be required in some installation conditions.

group

A configuration object in Oracle GoldenGate Veridata Web that is a logical container for one or more compare pairs (sets of related source and target tables) and their associated data source connections.

hash

A method of converting data into a unique numeric value (hash value) by substituting or transposing the data. If two hash values are different, then the degree of certainty that the two original input values are identical is extremely high (but not absolute). To be absolutely certain that values are identical, you can use a literal comparison.

in-flight

One of the possible states of a row during a comparison. The target row was out-of-sync in the initial comparison step, but was since updated and still do not match those of the source.

in-sync

One of the possible states of a row during comparison. The target row values match the source values that were retrieved during the initial comparison step.

initial comparison step, or row-hash step

The first step in a comparison, in which rows are retrieved from the source and target tables with a query and then compared. Target rows that appear to be out-of-sync are

stored in a maybe out-of-sync (MOOS) queue in memory. The rows will be assessed further by the confirmation step.

install

A program that is used on Windows platforms to install Oracle GoldenGate Veridata programs as a Windows service.

job

A unit of work that is linked to one or more compare groups that are processed together.

key, or key columns

A unique identifier that Oracle GoldenGate Veridata uses to order rows for comparison. This could be a primary key, a unique index, or a user-defined identifier.

latency

The difference in time between when a change is made to source data and when that change is reflected in the target data.

literal comparison

A comparison that is performed value for value, as opposed to a comparison that uses a hash.

Manager

Coordinator program of the Oracle GoldenGate Veridata Agent component. It coordinates requests sent to the agent process by the Oracle GoldenGate Veridata Server process.

Maybe-out-of-sync queue, or MOOS queue

A section of memory that stores rows that were determined to be out-of-sync in the initial comparison step, until they can be confirmed as in-sync or out-of-sync in the confirmation step.

Oracle GoldenGate Veridata Agent

Runs on the system where a source or target database resides. It fetches and returns blocks of rows, determines whether rows are out-of-sync, returns column-level detail, and executes requests on the database. The agent is started at the request of the Oracle GoldenGate Veridata Server and is controlled by the Oracle GoldenGate Veridata Manager process.

Oracle GoldenGate Veridata Server

The comparison engine that coordinates Oracle GoldenGate Veridata tasks, compares data, confirms out-of-sync data, and produces reports. Row sorting also can be performed by the server if the server-side sorting option is selected.

out-of-sync

The data values in a source row are not identical to those of the corresponding target row.

out-of-sync report

A report that is created at the conclusion of a comparison that provides summary and column-level details about out-of-sync rows and the operations that are needed to bring them back into synchronization.

partition

A selection criteria that is defined within Oracle GoldenGate Veridata Web or the vericom interface, which enables comparisons to be performed against a subset of the rows in a table or file.

persistently out-of-sync

One of the possible states of a row during comparison. During the confirmation step, the target values have not changed since the initial comparison step and still are not the same as those of the source.

profile

A configuration object in Oracle GoldenGate Veridata Web that is a stored set of global processing parameters, each containing unique settings for a specific purpose.

replication

The process of duplicating sets of data to one or more other locations in near real-time for such purposes as recovery or decision support.

repository

A collection of database objects that persists configuration information created by Oracle GoldenGate Veridata Web users to disk, saving it permanently as a user environment.

row partition

See partition.

row-hash step

See initial comparison step.

server

See Oracle GoldenGate Veridata Server.

server-side sorting

The sorting of source and target rows for comparison when performed by Oracle GoldenGate Veridata Server.

source

The database or location that is trusted to contain the most accurate and current version of data in an environment that contains replica versions of that data.

target

A database or location that contains a secondary or replica set of data that corresponds to a primary, or source, set of data.

validation

Performed by Oracle GoldenGate Veridata to test source and target table structures for compatibility.

vericom

The command line interface to Oracle GoldenGate Veridata.