Oracle® GoldenGate

Oracle Installation and Setup Guide

Version 10.4

October 2009

Oracle Installation and Setup Guide, version 10.4

Copyright © 1995, 2009 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 software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software 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 which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on 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. 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.

Contents

Chapter 1 System requirements and preinstallation instructions 5

Overview of GoldenGate for Oracle 5
Supported Platforms 5
Operating system requirements 5
Database requirements 8
Supported data types 11
Numeric data types 11
Character data types 11
Multi-byte character types 11
Binary data types 12
Date and timestamp data types 12
Large object data types 12
XML data types 13
User defined types (objects) 13
Other supported data types 15
Non-supported data types 15
Supported objects and operations for DML 15
Tables, views, and materialized views 15
Sequences 18
Supported objects and operations for DDL 18
Non-supported objects and operations for DML 19
Non-supported objects and operations for DDL 19
Oracle-reserved schemas 19
Oracle recycle bin 19
Supported and non-supported object names and case 20
Object names and owners 20
Case sensitivity 20
Supported characters 21
Non-supported characters 22

Chapter 2 Installing GoldenGate 23
Installation overview 23
Upgrades 23
New installations 23
Downloading GoldenGate 23
Setting ORACLE_HOME and ORACLE_SID 24
Setting library paths for dynamic builds on UNIX systems 25
Installing GoldenGate on Linux and UNIX 27
Installing into a UNIX or Linux cluster 27
Installing the GoldenGate files 27
Configuring Manager and other processes 27
Installing GoldenGate on Windows and Windows Cluster 28
Obtaining the Microsoft redistributable package 28
Installing GoldenGate into a Windows Cluster 28
Installing the GoldenGate files 28
Specifying a custom Manager name 28
Installing Manager as a Windows service 29
Adding GoldenGate as a Windows cluster resource 30
Configuring Manager and other processes 35
Chapter 3 Installing GoldenGate DDL support for an Oracle database 36
Overview of DDL support 36
Installing the DDL objects 37
Chapter 4 Preparing the database for GoldenGate 40
Ensuring ASM connectivity 40
Configuring character sets 40
Configuring the Oracle redo logs 41
Setting redo parallelism for Oracle 9*i* sources 41
Avoiding log-read bottlenecks 41
Mounting logs that are stored on other platforms 42
Ensuring data availability 42
Configuring GoldenGate to read archived logs only 43
Limitations of ALO mode 43
Configuring Extract for ALO mode 43
Managing the effects of row chaining 44
Adjusting cursors 44
Managing long-running transactions 45
Setting fetch options 46

Preparing tables for processing 48
Disabling triggers and cascade delete constraints 48
Assigning row identifiers 48
Getting the database to log key values 49
Limiting row changes in tables that do not have a key 50
Deferring constraint checking 51
Replicating Oracle TDE data 51
Ensuring correct handling of Oracle Spatial objects 52
Mapping the tables 52
Sizing the XML memory buffer 52
Handling triggers on the georaster tables 52
Consolidating fetching 55
Managing LOB caching 55
Additional requirements and procedures for Oracle RAC 55
General requirements 55
GoldenGate parameters settings for RAC 55
Special procedures on RAC 56

Chapter 5 Managing the Oracle DDL replication environment 57

Enabling and disabling the DDL trigger 57
Maintaining the DDL marker table 57
Deleting the DDL marker table 58
Maintaining the DDL history table 58
Deleting the DDL history table 58
Purging the DDL trace file 59
Applying database patches and upgrades when DDL support is enabled 59
Applying GoldenGate patches and upgrades when DDL support is enabled 59
Changing DDL object names after installation 60
Restoring an existing DDL environment to a clean state 62
Removing the DDL objects from the system 63

Chapter 6 Uninstalling GoldenGate 64

Uninstalling GoldenGate from UNIX 64
Uninstalling GoldenGate from Windows (non-cluster) 64
Uninstalling GoldenGate from Windows Cluster 65

Appendix 1 GoldenGate installed components 66

GoldenGate Programs and Utilities 66
GoldenGate subdirectories 68
Other GoldenGate files 70

GoldenGate checkpoint table 74

Index 75

CHAPTER 1

System requirements and preinstallation instructions

Overview of GoldenGate for Oracle

With GoldenGate for Oracle, you can replicate DML and DDL operations.

You can move data between similar or dissimilar supported Oracle versions, or you can move data between an Oracle database and a database of another type. GoldenGate supports the filtering, mapping, and transformation of data, unless otherwise noted in this documentation.

You can replicate DDL operations between similar Oracle databases. When DDL support is active, GoldenGate does not support the filtering, mapping, and transformation of data.

Supported Platforms

Databases

- Oracle 8i (DML support only)
- Oracle 9.1 and 9.2 (DML and DDL support)
- Oracle 10.1 and 10.2 (DML and DDL support)
- Oracle 11g (DML and DDL support)

Operating systems

To find out which GoldenGate builds are available for a specific combination of database version and operating system, go to http://support.goldengate.com. A valid user name and password are required to enter this site.

Operating system requirements

Memory requirements

The amount of memory that is required for GoldenGate depends on the number of concurrent processes that will be running.

- The GoldenGate GGSCI command interface fully supports up to 300 concurrent Extract and Replicat processes per instance of GoldenGate. An instance of GoldenGate equates to one Manager process, which is the main controller process.
- Each Extract and Replicat process needs approximately 25-55 MB of memory, or more depending on the size of the transactions and the number of concurrent transactions.

The GoldenGate cache manager takes advantage of the memory management functions of the operating system to ensure that GoldenGate processes work in a sustained and efficient manner. Within its cache, it makes use of modern virtual memory techniques by:

- Allocating and managing active buffers efficiently.
- Recycling old buffers instead of paging to disk, when possible.
- Paging less-used information to disk, when necessary.

The actual amount of physical memory that is used by any GoldenGate process is controlled by the operating system, not the GoldenGate program.

The cache manager keeps a GoldenGate process working within the soft limit of its global cache size, only allocating virtual memory (not physical memory) on demand. System calls to increase the cache size are made only as a last resort and, when used, are always followed by the release of virtual memory back to the system.

The system must have sufficient swap space for each GoldenGate Extract and Replicat process that will be running. To determine the required swap space:

1. Start up one Extract or Replicat.
2. Run GGSCI.
3. View the report file and find the line PROCESS VM AVAIL FROM OS (min).
4. Round up the value to the next full gigabyte if needed. For example, round up 1.76GB to 2 GB.
5. Multiply that value by the number of Extract and Replicat processes that will be running. The result is the maximum amount of swap space that could be required.

Disk requirements

- Assign the following free disk space:
  - 50 MB for the GoldenGate installation files. This includes space for the compressed download file and space for the uncompressed files. You can delete the download file after the installation is complete.
  - 40 MB for the working directories and binaries for each instance of GoldenGate that you are installing on the system. For example, to install two builds of GoldenGate into two separate directories, allocate 80 MB of space.
  - Additional disk space on any system that hosts GoldenGate trails, which contain the working data. The space that is consumed by the trails varies, depending on the volume of data that will be processed. A good starting point is 1 GB.
  - To install GoldenGate into a cluster environment, install the GoldenGate binaries and files on a shared file system that is available to all cluster nodes.
- To run GoldenGate for multiple Oracle instances on a Windows system, you must install an instance of GoldenGate for each one

Temporary disk requirements

By default, GoldenGate maintains data that it swaps to disk in the dirtmp sub-directory of the GoldenGate installation directory. The cache manager assumes that all of the free space on the file system is available. To avoid contention for disk space between GoldenGate and other applications, it is best practice to assign GoldenGate its own disk for its temporary writes. You can assign a directory by using the CACHEDIRECTORY option of the CACHEMGR parameter.

Oracle RAC requirements

- To install GoldenGate in an Oracle Real Application Cluster (RAC) environment, install GoldenGate on the shared drive(s) that are accessed by the RAC nodes. This allows you to start the GoldenGate processes from any of the nodes. If the node where the processes are running fails, the processing checkpoints are preserved in the installation directory, so you can start the processes on another node without modifying parameter files.
- All nodes in the RAC cluster must have synchronized system clocks. These clocks also must be synchronized with the clock on the system where Extract is executed. GoldenGate compares the local system's time to commit timestamps to make critical decisions. For information about synchronizing system clocks, consult www.ntp.org or your systems administrator. See also the IOLATENCY option of the THREADOPTIONS parameter in the *GoldenGate for Windows and UNIX Reference Guide*.

TCP/IP

- Configure the system to use TCP/IP services, including DNS.
- Configure the network with the host names or IP addresses of all systems that will be hosting GoldenGate processes and to which GoldenGate will be connecting. Host names are easier to use.
- GoldenGate requires the following unreserved and unrestricted TCP/IP ports:
  - One port for communication between the Manager process and other GoldenGate processes.
  - A range of ports for local GoldenGate communications: can be the default range starting at port 7840 or a customized range of up to 256 other ports.
- Keep a record of the ports you assigned to GoldenGate. You will specify them with parameters when configuring the Manager process.
- Configure your firewalls to accept connections through the GoldenGate ports.
- If possible, grant unrestricted FTP access to GoldenGate for transfers of data, parameters, and reports between source and target systems. Otherwise, provide for another transfer method. A secure transfer method is also required to resolve support cases.
- If possible, provide a connection between your source and target systems and a site where files can be staged for transfer to and from the GoldenGate Software FTP Support Site (ftp://support.goldengate.com).

Operating system privileges

- To install on Windows, the user installing GoldenGate must log in as Administrator.
- To install on UNIX, the user installing GoldenGate must have read and write privileges on the GoldenGate installation directory.
- The GoldenGate processes require an operating system user that has privileges to read, write, and delete files and subdirectories in the GoldenGate directory. In addition, the user for the Manager process requires privileges to control GoldenGate processes.
- The Extract process requires an operating system user that has read access to the transaction log files, both online and archived. On UNIX systems, that user must be a member of the group that owns the Oracle instance.
- It is recommended that these operating system users be dedicated to GoldenGate. Sensitive information might be available to anyone running a GoldenGate process, depending on how database authentication is configured.

Itanium requirements

To install GoldenGate on a Microsoft Itanium system, the vcredist_IA64.exe runtime library package must be installed. You can download this package from the Microsoft website. This package includes VisualStudio DLLs necessary for GoldenGate to operate on the Itanium platform. If these libraries are not installed, GoldenGate generates the following error.

```
"The application failed to initialize properly (0xc0150002). Click on Ok
to terminate the application.
```

Third-party programs

- Before installing GoldenGate on a Windows system, install and configure the Microsoft Visual C ++ 2005 SP1 Redistributable Package. **Make certain it is the SP1 version of this package, and make certain the get the right bit version for your server**. This package installs runtime components of Visual C++ Libraries. For more information, and to download this package, go to http://www.microsoft.com.
- GoldenGate fully supports virtual machine environments created with any virtualization software on any platform. When installing GoldenGate into a virtual machine environment, select a GoldenGate build that matches the database and the operating system of the virtual machine, not the host system. For example, on a Windows system with a RHAS 4.0 virtual machine running Oracle11g, you would install the GoldenGate RHAS 4.0 build for Oracle 11g, just as you would on an actual Linux machine.

Database requirements

Database configuration

- See Chapter 4 on page 40 for database configuration guidelines.

Database client

The full Oracle client must be used with GoldenGate so that the GoldenGate programs have access to the Oracle XDK libraries. Do not use Oracle Instant Client, which lacks those libraries. You can download the full client from Oracle's website.

Database user

- Create a database user that is dedicated to GoldenGate. It can be the same user for all of the GoldenGate processes that must connect to a database:
  - Extract (source database)
  - Replicat (target database)
  - Manager (source database, if using DDL support)
  - DEFGEN (source or target database)
- To preserve the security of your data, and to monitor GoldenGate processing accurately, do not permit other users, applications, or processes to log on or operate as the GoldenGate database user.
- If Oracle 10*g* Automatic Storage Management (ASM) is in use, GoldenGate requires a user for the Extract process to access the ASM instance. GoldenGate does not support using O/S authentication for the ASM user. You can use SYS user or any user with SYSDBA privileges in the ASM instance. See Table 3.
- Keep a record of the database users. They are required in the GoldenGate parameter files: the USERID parameter for the database user and the TRANLOGOPTIONS parameter with the ASMUSER and ASMPASSWORD options for the ASM user.

Table 1 Database user privileges — Oracle 8i

User privilege	Extract	Replicat
CREATE ANY TRIGGER	X[1]	
SELECT_CATALOG_ROLE	X	X
SELECT ON: sys.obj$ sys.user$ sys.tab$ sys.col$ sys.cdef$ sys.con$ sys.props$	X	X (sys.obj$ sys.user$ only)
CREATE SESSION, ALTER SESSION	X	X
RESOURCE	X	X[2]
CONNECT	X	X[3]
SELECT ANY TABLE or SELECT ON <owner.table>	X	X

Table 1 Database user privileges — Oracle 8i (continued)

User privilege	Extract	Replicat
INSERT, UPDATE, DELETE ON <target tables>		X
CREATE TABLE[4]		X

[1] Required to add update trigger with ADD TRANDATA command
[2] If RESOURCE cannot be granted to Replicat, use ALTER USER <user> QUOTA {<size> | UNLIMITED} ON <tablespace>, where <tablespace> represents all tablespaces that contain target objects.
[3] Required only if Replicat owns target objects or any PL/SQL procedures. If CONNECT cannot be granted, grant CREATE <object> for any object Replicat will need to create.
[4] Required if using ADD CHECKPOINTTABLE in GGSCI to use the database checkpoint feature.

Table 2 Database user privileges — Oracle 9i and later

User privilege	Extract	Replicat	Manager
CREATE SESSION, ALTER SESSION	X	X	
RESOURCE	X	X[1]	
CONNECT	X	X[2]	
SELECT ANY DICTIONARY	X	X	
FLASHBACK ANY TABLE or FLASHBACK ON <owner.table>	X		
SELECT ANY TABLE or SELECT ON <owner.table>	X	X	
INSERT, UPDATE, DELETE ON <target tables>		X	
CREATE TABLE[3]		X	
Privileges required to issue DDL operations to target tables (DDL support only).		X	
EXECUTE on DBMS_FLASHBACK package[4]	X		
GGS_GGSUSER_ROLE[5]	X		
DELETE ON GoldenGate DDL objects[6]			X
Oracle 10 g ASM privileges (see "")	X		

[1] If RESOURCE cannot be granted to Replicat, use ALTER USER <user> QUOTA {<size> | UNLIMITED} ON <tablespace>, where <tablespace> represents all tablespaces that contain target objects.
[2] Required only if Replicat owns target objects or any PL/SQL procedures. If CONNECT cannot be granted, grant CREATE <object> for any object Replicat will need to create.

[3] Required if using ADD CHECKPOINTTABLE in GGSCI to use the database checkpoint feature.
[4] GoldenGate must make a call to DBMS_FLASHBACK.GET_SYSTEM_CHANGE_NUMBER.
[5] Role for DML privileges on GoldenGate-owned DDL objects, if DDL support is used. Role is created during installation of those objects. User that installs this role must have SYSDBA privileges.
[6] Required only if using parameters that maintain the GoldenGate DDL database objects.

Table 3 Database user privileges — ASM instance

ASM password configuration[1]	Permitted user
ASM instance and the database share a password file	You can use the GoldenGate database user if you grant that user SYSDBA, or you can use any other database user that has SYSDBA privileges.
ASM instance and the database have separate password files	You can overwrite the ASM password file with the database password file, understanding that this procedure changes the SYS password in the ASM instance to the value that is contained in the database password file, and it also grants ASM access to the other users in the database password file. Save a copy of the ASM file before overwriting it.

[1] To view how the current ASM password file is configured, log on to the ASM instance and issue the following command in SQL*Plus:

```
SQL> SELECT name, value FROM v$parameter
WHERE name = 'remote_login_passwordfile';
```

Supported data types

Numeric data types

- NUMBER up to the maximum size permitted by Oracle
- BINARY FLOAT
- BINARY DOUBLE

Character data types

- CHAR
- VARCHAR2
- LONG
- NCHAR
- NVARCHAR2

Multi-byte character types

- NCHAR and NVARCHAR2 multi-byte character data types
- Multi-byte data stored in CHAR and VARCHAR2 columns

Limitations of support

- For GoldenGate to support multi-byte character data, the source and target databases must be identical. Transformation, filtering, and other manipulation are not supported.
- If replicating multi-byte CLOB data, set the Extract parameter TRANLOGOPTIONS to use the CONVERTUCS2CLOBS option. In a multi-byte database, CLOB data is represented in the redo log as AL16UTF16, which is compatible with UCS2 encoding. TRANLOGOPTIONS CONVERTUCS2CLOBS converts the data from UCS2 to the client character set.
- Multi-byte characters can be used with limitations in MAP and TABLE parameter statements such as string-based conversion functions and WHERE clauses. The code point for a multi-byte character must be represented within an escape sequence, for example "\u20ac." For more information, see the *GoldenGate for Windows and UNIX Administrator Guide* and the *GoldenGate for Windows and UNIX Reference Guide*.
- Multi-byte data is supported whether byte or character semantics are in use. If the source semantics setting is in bytes and the target is in characters, use the Replicat parameter SOURCEDEFS in your configuration, and place a DEFGEN-generated definitions file on the target. This configuration is required whether or not the source and target data definitions are identical. Replicat refers to the definitions file to determine the upper size limit for fixed-size character columns. For information about SOURCEDEFS and DEFGEN, see the *GoldenGate for Windows and UNIX Administrator Guide*.
- If the database has an NLS_NCHAR_CHARACTERSET value other than AL16UTF16, use the VARWIDTHNCHAR parameter to force NCHAR data to be written to the trail with 2-byte length information. For more information, see the *GoldenGate for Windows and UNIX Reference Guide*.

Binary data types

- RAW
- LONG RAW

Date and timestamp data types

- DATE
- TIMESTAMP (see non-supported data types for exclusions)

Large object data types

- CLOB
- NCLOB
- BLOB

Limitations of support

- SECUREFILE and BASICFILE are both supported.
- Store large objects out of row if possible.
- The use of OCILobWrite or DBMS_LOB functions to modify large objects is supported only for Oracle 9i databases and later.
- If a LOB is stored in-row, Extract captures its data from the redo log. If the LOB is stored out-of-row, Extract fetches its value from the database. If a value gets deleted before the fetch occurs, Extract writes a null to the trail. If a value gets updated before

a fetch, Extract writes the updated value. To prevent these inaccuracies, try to keep Extract latency low. The GoldenGate documentation provides guidelines for tuning process performance.

- SECUREFILE LOBs are always fetched from the database, whether or not they are stored in-row.
- When changing a SECUREFILE LOB from one storage to another (such as from ENCRYPT to DECRYPT), Oracle updates the whole table, and Extract captures those updates from the log. Therefore, it will appear as though Oracle updated all of the data blocks that are associated with the table.

  NOTE This also can happen if an ALTER TABLE command is issued to set a DEFAULT value to a column that has null values.

- To capture a large object from an Oracle 8i database, the associated row must be modified within the transaction. For example, you could use a stored procedure to set a column value to itself.
- If CLOB columns can store binary data, set the NLS_LANG system environment variable and the NLS_LANGUAGE database parameter to the same value.

- When the size of a large object exceeds 4K, GoldenGate stores the data in segments within the GoldenGate trail. The first 4K is stored in the base segment, and the rest is stored in a series of 2K segments. GoldenGate does not support filtering, column mapping, or manipulation for large objects of this size. Full GoldenGate functionality can be used for objects that are 4K or smaller.

XML data types

- XMLType for Oracle 9i and later

Limitations of support

- The source and target objects must be identical. Filtering and manipulation are not supported. However, you can map the XML representation of an object to a character column by means of a COLMAP clause in a TABLE or MAP statement.
- GoldenGate treats XMLType data as a LOB. There is no size limitation, but see "Large object data types" on page 12 for additional support and limitations.
- A table that contains XMLType columns must have one of the following: a primary key, column(s) with a unique constraint, or a unique index.

User defined types (objects)

GoldenGate supports user defined types (UDT) when the source and target objects have the same structure. The schema names can be different.

General limitations of support

- GoldenGate supports UDTs for Oracle 9.2.0 and later, with limitations described herein.
- Extract must fetch UDTs (except for object tables) from the database, so you should configure and use a snapshot for data consistency.

- Because a UDT must be fetched, a table that contains one must have one of the following: a primary key, column(s) with a unique constraint, or a unique index.
- GoldenGate does not support UDTs with the following embedded scalar types: LOB, CLOB, CFILE, BFILE, or INTERVAL_YM, INTERVAL_DS, and OPAQUE (with the exception of XMLType, which is supported).
- Object or relational tables where the key contains a UDT, or where a UDT is the only column, are not supported.
- The RMTTASK parameter does not support user-defined types (UDT).
- CHAR and VARCHAR attributes that contain unprintable characters are not supported.
- The GoldenGate DEFGEN, DDLGEN, and TRIGGEN utilities do not support UDTs.
- UDTs, including values inside object columns or rows, cannot be used within filtering criteria in TABLE or MAP statements, or as input or output for GoldenGate's column-conversion functions, SQLEXEC, or other built-in data-manipulation tools. Support is only provided for like-to-like Oracle source and targets.
- GoldenGate does not support REF types.

Limitations for collection types

- When data in a nested table is updated, the row that contains the nested table must be updated at the same time.
- When VARRAYS and nested tables are fetched, the entire contents of the column are fetched each time, not just the changes.

Limitations for object tables

- GoldenGate supports object tables in uni-directional and active-active configurations for Oracle 10g and later. Object tables are captured from the redo log; however, certain data types that are fetched from the database when in regular relational tables, such as LOBs and collection types, will also be fetched when in object tables. Similarly, current limitations that apply to collection types when in regular tables also apply to these types when in object tables.
- An Oracle object table can be mapped to a non-Oracle object table in a supported target database.
- A primary key must be defined on the root-level object attributes of the object table, and cannot include leaf-level attributes. If no key is defined, GoldenGate will use all viable columns as a pseudo-key.
- GoldenGate does not support the replication of DDL operations for an object table. This limitation includes the database object versioning that is associated with ALTERs of object tables.
- Synonyms are not supported for object tables or relational tables that contain object tables.

Limitations for spatial types

- GoldenGate supports SDO_GEOMETRY and SDO_TOPO_GEOMETRY, and SDO_GEORASTER (raster tables) for Oracle 10g and later.
- See additional configuration information for spatial types in "Ensuring correct handling of Oracle Spatial objects" on page 52.

Limitations for Oracle multimedia

- GoldenGate supports Oracle Multimedia ORDDicom object type.

Other supported data types

- ROWID
- VARRAY
- INTERVAL DAY and INTERVAL YEAR if the size of the target column is equal to, or greater than, that of the source.

Non-supported data types

- ANYDATA
- ANYDATASET
- ANYTYPE
- BFILE
- BINARY_INTEGER
- MLSLABEL
- PLS_INTEGER
- TIMEZONE_ABBR
- TIMEZONE_REGION
- URITYPE
- UROWID

Supported objects and operations for DML

Tables, views, and materialized views

GoldenGate supports the following DML operations made to regular tables, index-organized tables (ORGANIZATION INDEX clause of CREATE TABLE), clustered tables, and materialized views.

- INSERT
- UPDATE
- DELETE
- Associated transaction control operations

Limitations of support for regular tables

- GoldenGate supports tables that contain any number of rows up to 512 KB in length (except for LOBs). This size limitation mostly affects update operations on columns that are being used as a row identifier (primary or unique key, a key defined within the GoldenGate parameter file, or all columns if no key is defined). If a row identifier is updated, the 512 KB length must include not only the after image, but also the full before image, which is required to find the correct key on the target for the update.
- LOB columns are supported in their full size.
- GoldenGate supports the maximum number of columns per table that is supported by the database. GoldenGate supports the maximum column size that is supported by the database.

- GoldenGate supports tables that contain only one column, except when the column contains one of the following data types:
  - LOB
  - LONG
  - Nested table
  - User defined data type
  - VARRAY
  - XML
- GoldenGate supports tables with unused columns, but the support is disabled by default, and Extract abends on them. You can use the DBOPTIONS parameter with the ALLOWUNUSEDCOLUMN option to force Extract to generate a warning and continue processing. When using ALLOWUNUSEDCOLUMN, either the same unused column must exist in the target table, or a source definitions file must be created for Replicat with the DEFGEN utility. You can include the appropriate ALTER TABLE...SET UNUSED statements in a DDL replication configuration.
- GoldenGate supports tables with interval partitioning. Make certain that the WILDCARDRESOLVE parameter remains at its default of DYNAMIC.
- GoldenGate supports tables with virtual columns, but GoldenGate does not capture change data for these columns, because the database does not write it to the transaction log. You can use the FETCHCOLS option of the TABLE parameter to fetch the value of a virtual column. Replicat does not apply DML to a virtual column, even if the data for that column is in the trail, because the database does not permit DML on that type of column. Data from a source virtual column can be applied to a target column that is not a virtual column.
- In an initial load, all of the data is selected directly from the source tables, not the transaction log. Therefore, in an initial load, data values for all columns, including virtual columns, gets written to the trail or sent to the target, depending on the method that is being used. As when applying change data, however, Replicat does not apply initial load data to a target virtual column, because the database does not permit DML on that type of column.
- GoldenGate does not permit a virtual column to be used in a KEYCOLS clause in a TABLE or MAP statement.
- If a unique key includes a virtual column, and GoldenGate must use that key, the virtual column will be ignored. This might affect data integrity if the remaining columns do not enforce uniqueness. Fetching only provides an after value, and GoldenGate requires before and after values of keys.
- If a unique index is defined on any virtual columns, it will not be used.
- If a unique key or index contains a virtual column and is the only unique identifier on a table, GoldenGate must use all of the columns as an identifier to find target rows. Because a virtual column cannot be used in this identifier, it is possible that Replicat could apply operations containing this identifier to the wrong target rows.
- Tables created as EXTERNAL are not supported.
- A key cannot contain a column that is part of an invisible index.
- Tables created with table compression or OLTP table compression are not supported.
- GoldenGate supports the synchronization of TRUNCATE statements as part of the full DDL synchronization feature or as standalone functionality independent of full DDL synchronization. The standalone TRUNCATE feature supports the replication of TRUNCATE

TABLE, but no other DDL. The full DDL feature supports TRUNCATE TABLE, ALTER TABLE TRUNCATE PARTITION, and other DDL. To avoid errors from duplicate operations, only one of these features can be active at the same time. The GETTRUNCATES parameter controls the standalone TRUNCATE feature.

Limitations of support for views

- GoldenGate can replicate to a view as long as it is inherently updatable.GoldenGate supports capture from a table in the source database to an inherently updatable view in the target database.
- The structures of the table and the view must be identical.
- A key must be defined on the unique columns in the view. This is done by means of a KEYCOLS clause in the MAP statements.

Limitations of support for index-organized tables

- IOTs are supported for Oracle versions 10.2 and later.
- GoldenGate supports IOTs that are created with the MAPPING TABLE option, but it only captures changes made to the base IOT, not changes made to the mapping table. However, Oracle will maintain the mapping table on the target, if one is being used.
- IOTs that are stored in a compressed format are not supported (for example, in a compressed tablespace).

  NOTE A compressed IOT is different from an IOT that has key compression defined with the COMPRESS option. IOTs with key compression are supported.

- Because an IOT does not have a rowid, GoldenGate must fetch certain data types in an IOT from the database by key value, which increases the potential for "row not found" errors. GoldenGate provides the FETCHOPTIONS parameter to handle these errors. Data types that are fetched are:
  - BLOB
  - CLOB
  - NCLOB
  - XMLType
  - UDT
  - Nested table
  - VARRAY
- (Oracle 10g and later) TRUNCATES of an IOT where one partition is empty will not be captured.

Limitations of support for clustered tables

- Clustered tables are supported for Oracle versions 9i and later.
- Indexed and hash clusters are both supported.
- Encrypted and compressed clustered tables are not supported.

Limitations of support for materialized views

- Materialized views created WITH ROWID are not supported.
- The materialized view log can be created WITH ROWID.
- The source table must have a primary key.

- Truncates of materialized views are not supported. You can use a DELETE FROM statement.
- Some GoldenGate initial load methods do not support LOBs in a materialized view.
- For Replicat, the materialized view must be updateable.
- Full refreshes are supported for Oracle 10g and later.

Sequences

GoldenGate supports the replication of sequence values by means of the SEQUENCE parameter. GoldenGate ensures that the target sequence values will always be higher than those of the source (or equal to them, if the cache is 0).

NOTE DDL support for sequences (CREATE, ALTER, DROP, RENAME) is compatible with, but not required for, replicating sequence values. To replicate just sequence values, you *do not* need to install the GoldenGate DDL support environment. You can just use the SEQUENCE parameter.

Limitations of support for sequences

- The cache size and the increment interval of the source and target sequences must be identical.
- The cache can be any size, including 0 (NOCACHE).
- The sequence can be set to cycle or not cycle, but source and target must be set the same way.
- To add SEQUENCE to a configuration in which DDL support is enabled, you must re-install the GoldenGate DDL objects in INITIALSETUP mode.

Supported objects and operations for DDL

GoldenGate supports DDL for Oracle version 9i and later. DDL is not supported for earlier Oracle versions.

GoldenGate supports all DDL operations up to 2 MB in size on the following objects:

clusters	tables	triggers
functions	tablespaces	types
indexes	roles	views
packages	sequences	materialized views
procedure	synonyms	users

NOTE The actual size limit of the DDL support is approximate, because the size will not only include the statement text but also GoldenGate maintenance overhead that depends on the length of the object name, the DDL type, and other characteristics of keeping a DDL record internally.

Non-supported objects and operations for DML

- REF
- Tablespaces and tables created or altered with COMPRESS
- Synonyms
- /*+ APPEND */ hint (These are not directly captured or replicated, but you can configure Replicat to use an APPEND hint when applying data to a target table.)
- /*+ BUFFER */ hint
- Direct-path table loads
- Database Replay

Non-supported objects and operations for DDL

Oracle-reserved schemas

The following schema names are considered Oracle-reserved and must be excluded from the GoldenGate DDL configuration. GoldenGate will ignore these schemas.

Schema Name	Schema Name
ANONYMOUS	ORDPLUGINS
AURORA	ORDSYS
$JIS	OSE$HTTP$ADMIN
$UTILITY	OUTLN
$AURORA	PERFSTAT
$ORB	PUBLIC
$UNAUTHENTICATED	REPADMIN
CTXSYS	SYS
DBSNMP	SYSMAN
DMSYS	SYSTEM
DSSYS	TRACESVR
EXFSYS	WKPROXY
MDSYS	WKSYS
ODM	WMSYS
ODM_MTR	XDB
OLAPSYS	

Oracle recycle bin

Because of a known issue in Oracle 10g and later, the Oracle recycle bin must be turned off to support GoldenGate DDL replication. If the recycle bin is enabled, the GoldenGate DDL trigger session receives implicit recycle bin DDL operations that cause the trigger to fail.

When you install the GoldenGate DDL support objects, the script prompts you to permit it to purge the recycle bin, and then will do so automatically if permission is granted. However, you still must disable the recycle bin manually.

To turn off the recycle bin:

- Oracle 10*g* Release 2 and later: Set the RECYCLEBIN initialization parameter to OFF.
- Oracle 10*g* Release 1: Set the _RECYCLEBIN initialization parameter to FALSE.

Consult the database documentation for the correct syntax.

For more information about DDL support, including limitations of support and configuration guidelines, see the *GoldenGate for Windows and UNIX Administrator Guide*.

Supported and non-supported object names and case

The following will help you verify whether the name of a supported object qualifies or disqualifies it for inclusion in a GoldenGate configuration.

Object names and owners

Source and target object names must be fully qualified in GoldenGate parameter files, as in fin.emp.

Case sensitivity

If a database is case-sensitive, GoldenGate supports the case sensitivity of database names, owner names, object names, column names, and user names.

If a database is case-insensitive, or if it supports case-sensitivity but is configured to be case-insensitive, GoldenGate converts all names to upper case. The exception is Oracle 11g, where case-sensitive passwords are supported in GoldenGate input that requires passwords.

To preserve case-sensitivity

Case-sensitive names must be specified in GoldenGate parameter files exactly as they appear in the database. Enclose case-sensitive names in double quotes if the other database (the source or target of the case-sensitive objects) is not case-sensitive.

If replicating from a case-insensitive database to a case-sensitive database, the source object names must be entered in the Replicat MAP statements in upper case, to reflect the fact that they were written to the trail as uppercase by Extract.

For example:

```
MAP SALES.CUSTOMER, TARGET "Sales.Account";
```

Supported characters

GoldenGate supports alphanumeric characters in object names and the column names of key columns and non-key columns. GoldenGate also supports the following non-alphanumeric characters in columns that are not being used by GoldenGate as a key.

Table 4 Supported non-alphanumeric characters in object names and non-key column names[1]

Character	Description
~	Tilde
< >	Greater-than and less-than symbols
/	Forward slash
\	Backward slash
!	Exclamation point
@	At symbol
#	Pound symbol
$	Dollar symbol
%	Percent symbol
^	Carot symbol
( )	Open and close parentheses
_	Underscore
-	Dash
+	Plus sign
=	Equal symbol
|	Pipe
[ ]	Begin and end brackets
{ }	Begin and end curly brackets (braces)

[1] The type of key that is being used by GoldenGate depends on the definition of a given table and whether there are any overrides by means of a KEYCOLS clause. GoldenGate will use a primary key, if available, or a unique key/index (selection is dependent on the database). In the absence of those definitions, all columns of the table are used, but a KEYCOLS clause overrides all existing key types. For columns that are being used by GoldenGate as a key, the characters in the names must be valid for inclusion in a WHERE clause. This list is all-inclusive; a given database platform may or may not support all listed characters.

Non-supported characters

GoldenGate does not support the following characters in object or column names:

Table 5 Non-supported characters in object and column names[1]

Character	Description
&	Ampersand
*	Asterisk
?	Question mark
:	Colon
;	Semi-colon
,	Comma
‘ ’	Single quotes
“ ”	Double quotes
‘	Accent mark (Diacritical mark)
.	Period
	Space

[1] This list is all-inclusive; a given database platform may or may not support all listed characters.

CHAPTER 2

Installing GoldenGate

Installation overview

These instructions are for installing GoldenGate for the first time. Installing GoldenGate installs all of the components required to run and manage GoldenGate processing (exclusive of any components required from other vendors, such as drivers or libraries) and it installs the GoldenGate utilities. The installation process takes a short amount of time.

Upgrades

To upgrade GoldenGate from one version to another, follow the instructions on the GoldenGate support site at http://support.goldengate.com.

New installations

To install GoldenGate for the first time, the following steps are required:

- Downloading GoldenGate
- Setting ORACLE_HOME and ORACLE_SID
- Setting library paths for dynamic builds
- Installing the software

> **NOTE** Before proceeding, make certain that you have reviewed the System Requirements.

Downloading GoldenGate

1. Navigate to http://support.goldengate.com.
2. In the navigation bar, select Downloads.
3. In the navigation bar, select the platform.
4. Select the operating system and database.
5. Locate the correct GoldenGate build.
6. Click Download to transfer the software to your system.

Setting ORACLE_HOME and ORACLE_SID

Make certain that the ORACLE_HOME and ORACLE_SID system environment variables are set to the correct Oracle instance. The GoldenGate processes refer to them when connecting to the database.

To specify Oracle variables on UNIX-based systems

- If there is one instance of Oracle on the system, you only need to set ORACLE_HOME and ORACLE_SID at the system level. If you cannot set them that way, use the following SETENV statements in the parameter file of every Extract and Replicat group that will be connecting to the instance.

  ```
  SETENV (ORACLE_HOME = "<path to Oracle home location>")
  SETENV (ORACLE_SID = "<SID>")
  ```

  These parameters override the system settings and allow the GoldenGate process to set the variables at the session level when it connects to the database.

- If there are multiple Oracle instances on the system with Extract and Replicat processes connecting to them, you will need to use a SETENV statement in the parameter file of each process group and point it to the correct instance. For example, the following shows parameter files for two Extract groups, each processing from a different Oracle instance.

  Group 1:

  ```
  EXTRACT ora9a
  SETENV (ORACLE_HOME = "/home/oracle/ora9/product")
  SETENV (ORACLE_SID = "ora9a")
  USERID ggsa, PASSWORD ggsa
  RMTHOST sysb
  RMTTRAIL /home/ggs/dirdat/rt
  TABLE hr.emp;
  TABLE hr.salary;
  ```

  Group 2:

  ```
  EXTRACT ora9b
  SETENV (ORACLE_HOME = "/home/oracle/ora9/product")
  SETENV (ORACLE_SID = "ora9b")
  USERID ggsb, PASSWORD ggsb
  RMTHOST sysb
  RMTTRAIL /home/ggs/dirdat/st
  TABLE fin.sales;
  TABLE fin.cust;
  ```

To specify Oracle variables on Windows systems

- If there is one instance of Oracle on the system, the Registry settings for ORACLE_HOME and ORACLE_SID should be sufficient for GoldenGate. If those settings are incorrect in the Registry and cannot be changed, you can set an override as follows.
  - On the desktop or Start menu (depending on Windows version), right-click **My Computer**, and then select **Properties**.
  - In **Properties**, click the **Advanced** tab.

- Click **Environment Variables**.
- Under **System Variables**, click **New**.
- For **Variable Name**, type ORACLE_HOME.
- For **Variable Value**, type the path to the Oracle binaries.
- Click **OK**.
- Click **New** again.
- For **Variable Name**, type ORACLE_SID.
- For **Variable Value**, type the instance name.
- Click **OK**.

- If there are multiple Oracle instances on the system with Extract and Replicat processes connecting to them, do the following.
  - Use the preceding procedure (single Oracle instance on system) to set the ORACLE_HOME and ORACLE_SID system variables to the first Oracle instance.
  - Start all of the GoldenGate processes that will connect to that instance.
  - Repeat the procedure for the next Oracle instance, but this time **Edit** the existing ORACLE_HOME and ORACLE_SID variables to specify the new information.
  - Start the GoldenGate processes that will connect to that instance.
  - Repeat the **Edit** and startup procedure for the rest of the Oracle instances.

Setting library paths for dynamic builds on UNIX systems

As of version 10, GoldenGate uses shared libraries. When installing GoldenGate on a UNIX system, the following must be true *before running GGSCI or any GoldenGate process*.

1. When connecting to the database locally, all of the following must have the same bit type, either all 32-bit, all 64-bit, or all IA64:
   - GoldenGate version
   - Oracle library versions
   - Database versions
2. When connecting using SQL*Net, the Oracle client library and the GoldenGate build must match. This means that the Oracle version, the bit type (32-bit, 64-bit, IA64) and the operating system version all must match. If you are using the TRANLOGOPTIONS parameter with the LOGSOURCE option and connecting to transaction logs from a different operating system, the Oracle versions must also be the same.
3. Make certain that the database libraries are added to the system's shared-library environment variables. This procedure is usually performed at database installation time. Consult your Database Administrator if you have any questions.
4. If you will be running a GoldenGate program from outside the GoldenGate installation directory on a UNIX system:
   - (Optional) Add the GoldenGate installation directory to the PATH environment variable.
   - (Required) Add the GoldenGate installation directory to the shared-libraries environment variable.

For example, given a GoldenGate installation directory of /ggs/10.0, the second command in the following table requires these variables to be set:

Command	Requires GG libraries in environment variable?
$ ggs/10.0 > ./ggsci	No
$ ggs > ./10.0/ggsci	Yes

To set the variables in Korn shell

```
PATH=<installation directory>:$PATH
export PATH
<shared libraries variable>=<absolute path of installation directory>:$<shared libraries variable>
export <shared libraries variable>
```

To set the variables in Bourne shell

```
export PATH=<installation directory>:$PATH
export <shared libraries variable>=<absolute path of installation directory>:$<shared libraries variable>
```

To set the variables in C shell

```
setenv PATH <installation directory>:$PATH
setenv <shared libraries variable> <absolute path of installation directory>:$<shared libraries variable>
```

Where: <shared libraries variable> is one of the following:

UNIX/Linux library path variables per platform

Platform[1]	Environment variable
◆ IBM AIX ◆ IBM z/OS	LIBPATH
HP-UX	SHLIB_PATH
◆ Sun Solaris ◆ HP Tru64 (OSF/1) ◆ LINUX	LD_LIBRARY_PATH

[1] A specific platform may or may not be supported by GoldenGate for your database. See the Systems Requirements for supported platforms.

Example

```
export LD_LIBRARY_PATH=/ggs/10.0:$LD_LIBRARY_PATH
```

NOTE To view the libraries that are required by a GoldenGate process, use the ldd <process> shell command before starting the process. This command also shows an error message for any that are missing.

Installing GoldenGate on Linux and UNIX

Installing into a UNIX or Linux cluster

- To install GoldenGate into a cluster environment, install the GoldenGate binaries and files on a file system that is available to all cluster nodes, according to the directions that follow.
- After installing GoldenGate, configure the GoldenGate Manager process within the cluster application, as directed by the cluster documentation, so that GoldenGate will fail over properly with the other applications.

Installing the GoldenGate files

1. FTP the file in binary mode to the system and directory where you want GoldenGate to be installed.
2. Extract the gzipped tar file (use the gzip or tar options appropriate for your system). The files are placed in the current directory. If gzip is not installed, unzip the file on a Windows system by using WinZip or an equivalent compression product, and then FTP the file in binary format to the installation machine.

   ```
   gzip -dc <filename>.tar.gz | tar -xvof -
   ```

   This is an example:

   ```
   gzip -dc sun29_ora102_v9527_007.tar.gz | tar -xvof -
   ```

3. Run the command shell and change directories to the new GoldenGate directory.
4. From the GoldenGate directory, run the GGSCI program.

   ```
   GGSCI
   ```

5. In GGSCI, issue the following command to create the GoldenGate working directories.

   ```
   CREATE SUBDIRS
   ```

6. Issue the following command to exit GGSCI.

   ```
   EXIT
   ```

Configuring Manager and other processes

- To use GoldenGate, you must configure the Manager process. You must specify a TCP/IP port for Manager to use, and you can specify optional parameters that control dynamic port assignments, trail file maintenance, and other properties.
- To begin using GoldenGate, you need to create and configure at least one Extract and Replicat group. Your instructions for these groups determine which data to capture and replicate, and how that data is processed.
- To configure these processes, and to customize GoldenGate, see the *GoldenGate for Windows and UNIX Administrator Guide*.

Installing GoldenGate on Windows and Windows Cluster

Obtaining the Microsoft redistributable package

- Before installing GoldenGate on a Windows system, install and configure the Microsoft Visual C ++ 2005 SP1 Redistributable Package. **Make certain it is the SP1 version of this package, and make certain the get the right bit version for your server**. This package installs runtime components of Visual C++ Libraries. For more information, and to download this package, go to http://www.microsoft.com.

Installing GoldenGate into a Windows Cluster

1. Log into one of the nodes in the cluster.
2. For the GoldenGate installation location, choose a drive that is a resource within the same cluster group that contains the database instance.
3. Ensure that this group is owned by the cluster node that you are logging into.
4. Install GoldenGate according to the following instructions.

Installing the GoldenGate files

1. Unzip the downloaded file(s) using PKUNZIP or WinZip.
2. Move the files in binary mode to a folder on the drive where you want to install GoldenGate. *Do not* install GoldenGate into a folder that contains spaces in its name, for example "GoldenGate Software." GoldenGate relies on path names, but the operating system does not support path names that contain spaces, whether or not they are within quotes.
3. From the GoldenGate folder, run the GGSCI program.

   ```
   GGSCI
   ```

4. In GGSCI, issue the following command to create the GoldenGate working directories.

   ```
   CREATE SUBDIRS
   ```

5. Issue the following command to exit GGSCI.

   ```
   EXIT
   ```

Specifying a custom Manager name

You must specify a custom name for the Manager process if either of the following is true:

- you want to use a name for Manager other than the default of GGSMGR.
- there will be multiple Manager processes running as Windows services on this system, such as one for the GoldenGate replication software and one for GoldenGate Veridata. Each Manager on a system must have a unique name. Before proceeding further, verify the names of any local Manager services.

To specify a custom Manager name

1. From the directory that contains the Manager program, run GGSCI.
2. Issue the following command.

   ```
   EDIT PARAMS ./GLOBALS
   ```

3. In the file, add the following line, where <name> is a one-word name for the Manager service.

   ```
   MGRSERVNAME <name>
   ```

4. Save the file. The file is saved automatically with the name GLOBALS, *without a file extension*. Do not move this file. It is referenced during installation of the Windows service and during data processing.

Installing Manager as a Windows service

By default, Manager is not installed as a service and can be run by a local or domain account. However, when run this way, Manager will stop when the user logs out. When you install Manager as a service, you can operate it independently of user connections, and you can configure it to start manually or at system start-up. Installing Manager as a service is required on a Windows Cluster, but optional otherwise.

To install Manager as a Windows service

1. (Recommended) Log on as the system administrator.
2. Click **Start** > **Run**, and type **cmd** in the **Run** dialog box.
3. From the directory that contains the Manager program that you are installing as a service, run the **install** program with the following syntax:

   ```
   install <option> [...]
   ```

   Where: <option> is one of the following:

Table 6 INSTALL options

Option	Description
ADDEVENTS	Adds GoldenGate events to the Windows Event Manager. By default, GoldenGate errors are generic. To produce more specific error content, copy the following files from the GoldenGate installation directory to the SYSTEM32 directory. category.dll ggsmsg.dll

Table 6 INSTALL options (continued)

Option	Description
ADDSERVICE	Adds Manager as a service by the name specified in the GLOBALS file, if one exists, or by the default of GGSMGR. ADDSERVICE configures the service to run as the Local System account, the standard for most Windows applications because the service can be run independently of user logins and password changes. To run Manager as a specific account, use the USER and PASSWORD options.[1] The service is installed to start at system boot time (see AUTOSTART). To start it after installation, either reboot the system, or start the service manually from the Services applet of the Control Panel.
AUTOSTART	Specifies that the service created with ADDSERVICE is to be started at system boot time. This is the default unless MANUALSTART is used.
MANUALSTART	Specifies that the service created with ADDSERVICE is to be started manually through GGSCI, a script, or the Services applet of the Control Panel. The default is AUTOSTART.
USER <name>	Specifies a domain user account for executing Manager. For <name>, include the domain name, a backward slash, and the user name, for example HEADQT\GGSMGR. By default, the Manager service is installed to use the Local System account.
PASSWORD <password>	Specifies the password for the user specified with USER.

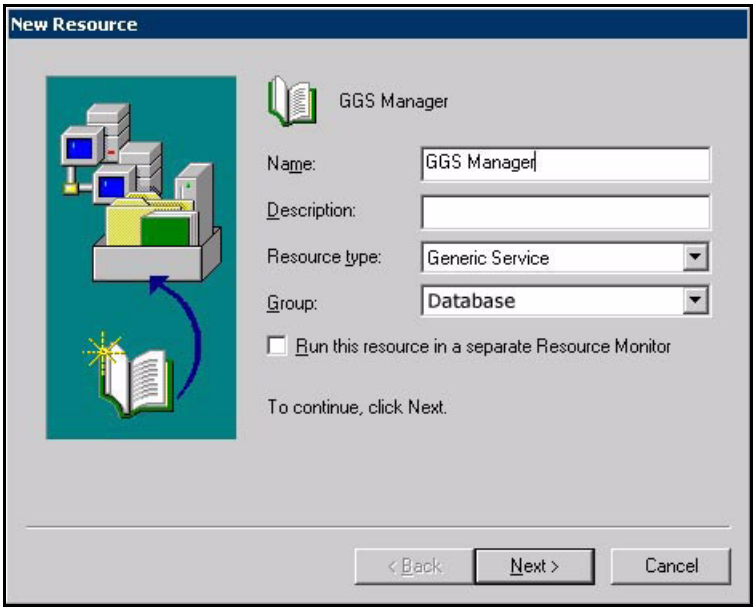
[1] A user account can be changed by selecting the Properties action from the Services applet of the Windows Control Panel.

Adding GoldenGate as a Windows cluster resource

If you installed GoldenGate into a cluster, follow these instructions to establish GoldenGate as a cluster resource and configure the Manager service correctly on all nodes.

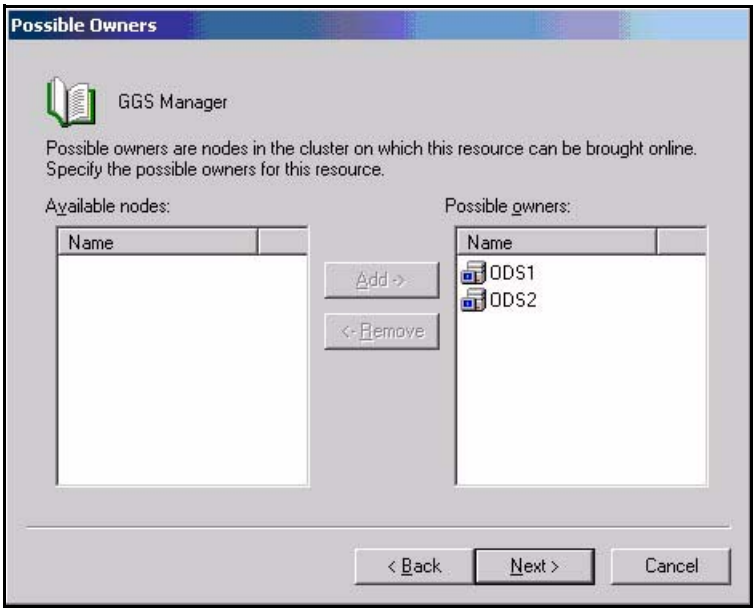
1. In the Cluster Administrator, select **File>New>Resource**.

2. In the New Resource dialog box, provide a descriptive name for the GoldenGate Manager (need not be its actual name). For Resource Type, select Generic Service. For Group, select the group that contains the database instance to which GoldenGate will connect.



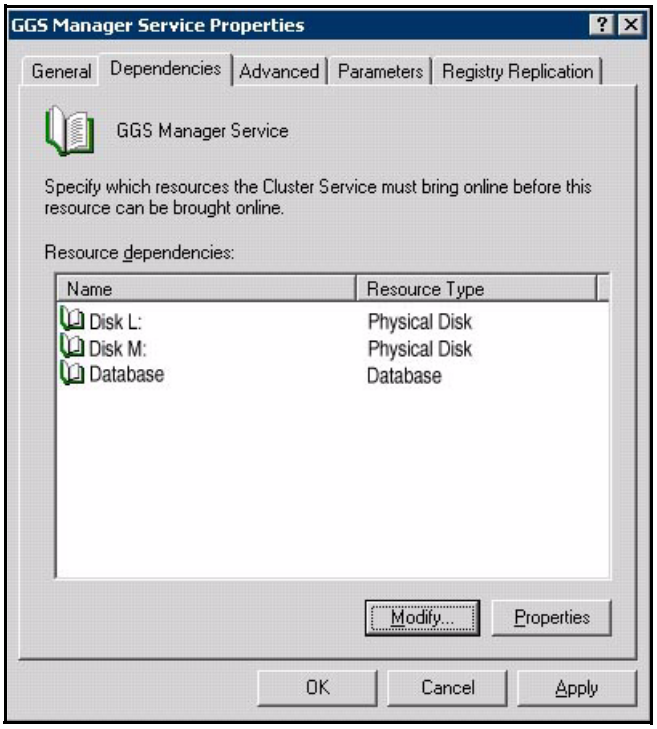
3. Click **Next**.

4. In the Possible Owners dialog box, select the nodes on which GoldenGate will run.



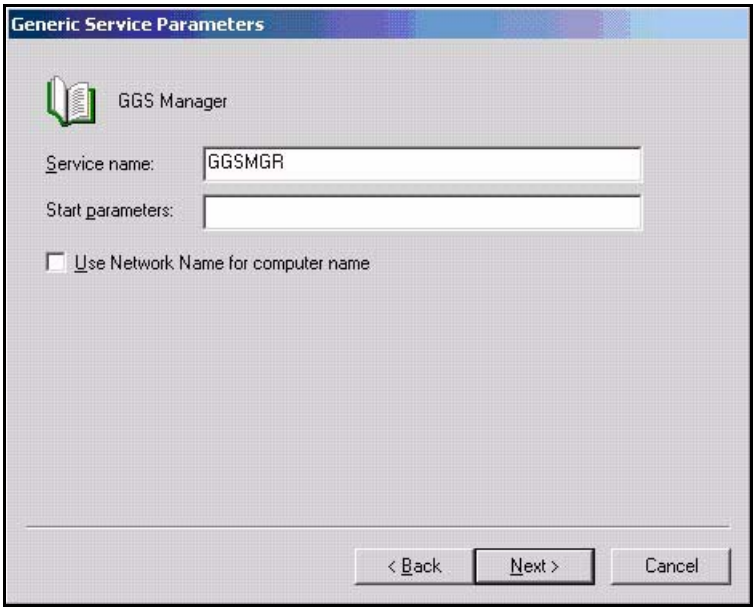
5. Click **Next**.

6. In the GGS Manager Service Properties dialog box, click the Dependencies tab, and add the following to the Resource dependencies list:
   - The database resource group (in this example, it is "Database")
   - The disk resource containing the GoldenGate directory
   - The disk resource containing the database transaction log files
   - The disk resource containing the database transaction log backup files



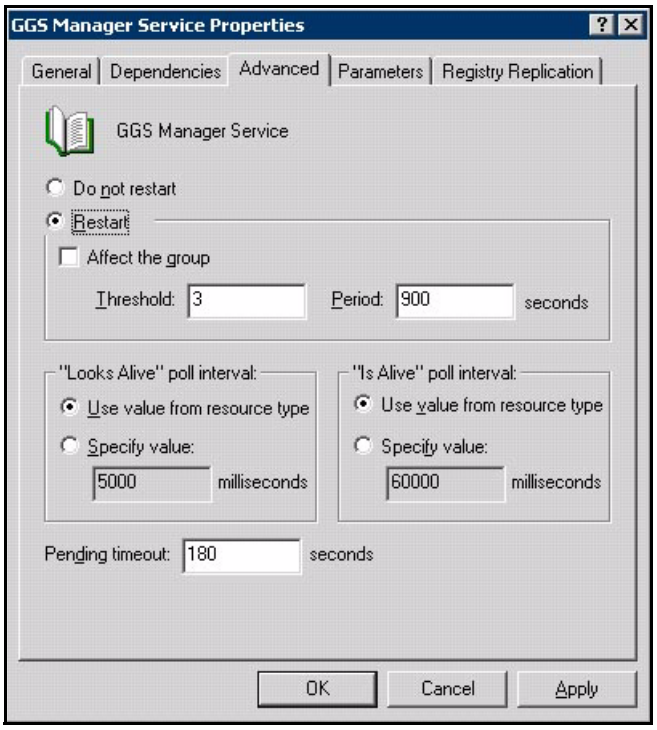
7. Click **Apply**, then **OK**.

8. In the Generic Service Parameters dialog box, type either the default Manager service name of GGSMGR or, if applicable, the custom name specified in the GLOBALS file.



9. Click **Next**.
10. Click **Finish** to exit the wizard.
11. In the Cluster Administrator tree, right-click the Manager resource and select Properties.

12. Click the Advanced tab, and deselect Affect the Group. This is a recommendation, but you can configure it as needed for your environment.



13. Click **Apply**.
14. Bring the cluster resource online to verify that it was installed correctly.
15. Take the resource offline again.
16. Move the group to the next node in the cluster. When the group has been successfully moved to the second node, the Manager resource should still be offline.
17. Log onto the second node.
18. Install GoldenGate Manager as a service on this node by running the **install** program as you did on the previous node. If you created a custom name for Manager in the GLOBALS file, that name will be used.
19. Bring the resource online to verify that it is running correctly on this node.
20. Repeat steps 18 through 22 for each additional node in the cluster.

Configuring Manager and other processes

- To use GoldenGate, you must configure the Manager process. You must specify a TCP/IP port for Manager to use, and you can specify optional parameters that control dynamic port assignments, trail file maintenance, and other properties.
- To begin using GoldenGate, you need to create and configure at least one Extract and Replicat group. Your instructions for these groups determine which data to capture and replicate, and how that data is processed.
- To configure these processes, and to customize GoldenGate, see the *GoldenGate for Windows and UNIX Administrator Guide*.

CHAPTER 3

Installing GoldenGate DDL support for an Oracle database

Overview of DDL support

This chapter contains instructions for installing the objects that support GoldenGate DDL replication for Oracle. To configure GoldenGate to capture and replicate DDL, see the *GoldenGate for Windows and UNIX Administrator Guide*.

NOTE DDL support for sequences (CREATE, ALTER, DROP, RENAME) is compatible with, but not required for, replicating sequence values. To replicate just sequence values, you *do not* need to install the GoldenGate DDL support environment. You can just use the SEQUENCE parameter.

To install the GoldenGate DDL environment, you will be installing the database objects shown in Table 7.

WARNING Do not include any GoldenGate-installed DDL objects in a DDL parameter, in a TABLE parameter, or in a MAP parameter, nor in a TABLEEXCLUDE or MAPEXCLUDE parameter. Make certain that wildcard specifications in those parameters do not include GoldenGate-installed DDL objects. These objects must not be part of the GoldenGate configuration, but the Extract process must be aware of operations on them, and that is why you must not explicitly exclude them from the configuration with an EXCLUDE, TABLEEXCLUDE, or MAPEXCLUDE parameter statement.

Table 7 DDL synchronization objects

Object	Purpose	Default name
DDL marker table	Stores DDL information. This table only receives inserts.	GGS_MARKER
Sequence on marker table	Used for a column in the marker table.	GGS_DDL_SEQ
DDL history table	Stores object metadata history. This table receives inserts, updates, deletes.	GGS_DDL_HIST

Table 7 DDL synchronization objects (continued)

Object	Purpose	Default name
Object ID history table	Contains object IDs of configured objects.	GGS_DDL_HIST_ALT
DDL trigger	Fires on DDL operations. Writes information about the operation to the marker and history tables. Installed with the trigger are some packages.	GGS_DDL_TRIGGER_BEFORE
DDL schema	Contains the DDL synchronization objects.	None; must be specified during installation and in the GLOBALS file.
User role	Establishes the role needed to execute DDL operations.	GGS_GGSUSER_ROLE
Internal setup table	Database table for internal use only.	GGS_SETUP
ddl_pin	Pins DDL tracing, the DDL package, and the DDL trigger for performance improvements.	ddl_pin
ddl_cleartrace.sql	Removes the DDL trace file.	ddl_cleartrace.sql
ddl_status.sql	Verifies that the GoldenGate DDL objects are installed	ddl_status.sql
marker_status.sql	Verifies that the marker table is installed.	marker_status.sql
ddl_tracelevel.sql	Sets the level for DDL tracing.	ddl_tracelevel.sql

Installing the DDL objects

1. Choose a GoldenGate schema or another schema for the DDL objects.
2. Grant the following permission on the GoldenGate schema.

   ```
   GRANT EXECUTE ON UTL_FILE TO <schema>;
   ```

3. Choose a tablespace for the DDL objects that can accommodate growth of the GGS_DDL_HIST and GGS_MARKER tables. The GGS_DDL_HIST table, in particular, will grow in proportion to overall DDL activity. If the tablespace that contains these objects fills up, no DDL operations can be issued on the database, and the business applications will pause.

4. Open the GLOBALS file in the home directory of this instance of GoldenGate. If a GLOBALS file does not exist, create one. For information on creating or editing a GLOBALS file, see the *GoldenGate for Windows and UNIX Administrator Guide*.
5. Specify the name of the schema by adding the following parameter to the GLOBALS file.

   GGSCHEMA <schema_name>

6. (Optional) To change the names of other objects listed in Table 7, *the changes must be made now, before proceeding with the rest of the installation. Otherwise, you will need to stop GoldenGate DDL processing and reinstall the DDL objects.* It is recommended that you accept the default names of the database objects. To change any name in Table 7 (except the schema), do one or both of the following:
   - All name changes must be recorded in the params.sql script. Edit this script and change the appropriate parameters. Do not run this script.
   - Some name changes must also be listed in the GLOBALS file. These are listed in Table 8. The correct parameters to use are listed in the Parameter column.

Table 8 GLOBALS parameters for changing DDL object names

Object	Parameter
Marker table	MARKERTABLE <new_table_name>[1]
History table	DDLTABLE <new_table_name>

[1] Do not qualify the name of any of these tables. The schema name for these table must be either the user specified with GGSCHEMA or the current user, if GGSCHEMA is not specified in GLOBALS.

7. Disable the Oracle recycle bin. The recycle bin must be disabled to support GoldenGate DDL processing. To turn off the recycle bin:
   - Oracle 10*g* Release 2 and later: Set the RECYCLEBIN initialization parameter to OFF.
   - Oracle 10*g* Release 1: Set the _RECYCLEBIN initialization parameter to FALSE.

   Consult the database documentation for the correct syntax.
8. Change directories to the GoldenGate installation directory.
9. Exit all Oracle sessions, including those in SQL*Plus, those open within business applications, those held open by the GoldenGate processes, any tools that use Oracle, and so forth. Prevent the start of any new sessions.
10. Run SQL*Plus and log in as a user that has SYSDBA privileges. This privilege is required to install the DDL trigger. The trigger will be installed in the SYS schema, which is required by Oracle, and all other DDL synchronization objects will be installed in the schema you created in step 1.
11. Run the marker_setup script. This script installs support for the GoldenGate marker system, required for DDL support. You will be prompted for the name of the GoldenGate schema.
12. Run the ddl_setup script. You will be prompted to:
    - Close any open Oracle sessions. Any open sessions are listed in the prompt.
    - Specify the name of the DDL schema from step 1.

- Specify the installation mode: To install DDL objects the first time, use the INITIALSETUP mode, which assumes no GoldenGate DDL objects exist and drops them if they do. If the DDL objects exist and you want to reinstall them, but preserve any DDL history, use the procedure in "Restoring an existing DDL environment to a clean state" on page 62.
- Grant the script permission to purge the recycle bin (Yes or No). Yes purges the recycle bin and completes the installation. No causes the script to return an error.

  NOTE This does not disable the recycle bin, but only purges it.

13. Run the role_setup script. This script drops and creates the role needed for DDL synchronization. It grants DML permissions on the GoldenGate DDL objects.
14. Grant the role to all GoldenGate Extract users. You may need to make multiple grants if the processes have different user names.
15. Run the ddl_enable.sql script to enable the DDL trigger.

To install and use the optional performance tool

To improve the performance of the DDL trigger, make the ddl_pin script part of the database startup. It must be invoked with the GoldenGate DDL user name, as in:

```
SQL> @ddl_pin <DDL_user>
```

This script pins the PL/SQL package that is used by the trigger into memory. If executing this script from SQL*Plus, connect as SYSDBA from the GoldenGate home directory. This script relies on the Oracle dmbs_shared_pool system package, so install that package before using ddl_pin.

CHAPTER 4

Preparing the database for GoldenGate

Ensuring ASM connectivity

To ensure that GoldenGate is able to connect to an ASM instance, do the following.

- Make certain that the ASM instance is listed in the tnsnames.ora file.
- Make certain that the Oracle listener is listening for new connections to the ASM instance. The listener.ora file should contain an entry similar to the following (specifically, the second SID_DESC).

```
SID_LIST_LISTENER_DARAN =
    (SID_LIST =
        (SID_DESC =
            (SID_NAME = PLSExtProc)
            (ORACLE_HOME = /rdbms/oracle/ora1012r/64)
            (PROGRAM = extproc)
        )
        (SID_DESC =
        (ORACLE_HOME = /rdbms/oracle/ora1012r/64)
        (SID_NAME = +ASM1)
        )
    )
```

Configuring character sets

To ensure accurate character representation from one database to another, the following must be true:

- The character set of the target database must be a superset of the character set of the source database.
- If your client applications use different character sets, the database character set must be a superset of the character sets of the client applications. In this configuration, every character is represented when converting from a client character set to the database character set.
- For more information, refer to Oracle's *Database Globalization Support Guide*.

To view globalization settings

To determine the globalization settings of the database and whether it is using byte or

character semantics, use the following commands in SQL*Plus:

```
SHOW PARAMETER NLS_LANGUAGE
SHOW PARAMETER NLS_TERRITORY
SELECT name, value$ from SYS.PROPS$ WHERE name = 'NLS_CHARACTERSET';
SHOW PARAMETER NLS_LENGTH_SEMANTICS
```

To view globalization settings from GGSCI

The VIEW REPORT <group> command in GGSCI shows the current database language and character settings and indicates whether or not NLS_LANG is set.

To set NLS_LANG

1. Set the NLS_LANG parameter according to the documentation for your database version and operating system. In UNIX, you can set NLS_LANG through the operating system or by using a SETENV parameter in the Extract and Replicat parameter files. For best results, set NLS_LANG from the parameter file, where it is less likely to be changed than at the system level.

   NLS_LANG must be set in the format of:

   ```
   <NLS_LANGUAGE>_<NLS_TERRITORY>.<NLS_CHARACTERSET>
   ```

   This is an example in UNIX, using the SETENV parameter in the GoldenGate parameter file:

   ```
   SETENV (NLS_LANG = “AMERICAN_AMERICA.AL32UTF8”)
   ```

2. Stop and then start the GoldenGate Manager process so that the GoldenGate processes recognize the new variable.

Configuring the Oracle redo logs

When operating in its normal mode, GoldenGate reads the online logs by default, but will read the archived logs if an online log is not available.

NOTE You can also configure GoldenGate to read exclusively from the archived logs. See page 43.

To ensure the continuity and integrity of processing when GoldenGate is reading from the online logs, configure the logs as follows.

Setting redo parallelism for Oracle 9*i* sources

If using GoldenGate for an Oracle 9*i* source database, set the _LOG_PARALLELISM parameter to 1. GoldenGate does not support values higher than 1.

Avoiding log-read bottlenecks

When GoldenGate captures data from the redo logs, I/O bottlenecks can occur because Extract is reading the same files that are being written by the database. Performance degradation is increased the more Extract processes there are that read the same logs. You can:

- Try using faster drives and a faster controller. Both Extract and the database logging mechanism will be faster on a faster I/O system.
- See if the logs are on RAID 0+1. Avoid RAID 5, which performs checksums on every block written and is not a good choice for high levels of continuous I/O. For more information, see Oracle's documentation or search related websites.

Mounting logs that are stored on other platforms

If the online and archived redo logs are stored on a different platform from the one the installed Extract is built for, do the following:

- NFS-mount the archive files.
- Map the file structure to the source Extract system's Windows or UNIX file structure by using the LOGSOURCE and PATHMAP options of the Extract parameter TRANLOGOPTIONS. See the reference documentation of this parameter for currently supported LOGSOURCE platforms.

> **NOTE** If used, put the TRANLOGOPTIONS statement on one line. Do not use ampersand (&) line terminators to split it into multiple lines.

Ensuring data availability

For best results, enable archive logging and keep the archived logs on the system for the longest time possible. Extract requires access to the log that contains the beginning of the oldest open transaction and all logs thereafter. The archives provide a secondary data source should the online logs recycle before Extract is finished with them.

If you cannot enable archive logging, configure the online logs to retain enough data so that Extract can capture what it needs before the logs recycle. Allow for Extract backlogs caused by network outages and other external factors, as well as long-running transactions.

The recommended retention period is at least 24 hours worth of transaction data, including both online and archived information (if enabled). You might need to do some testing to determine the best retention time given your data volume and business requirements.

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

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

Purging log archives

Make certain not to use backup or archive options that cause old archive files to be overwritten by new backups. It is best practice for any new backups to be separate files with different names from older ones. This ensures that if Extract looks for a particular log, it will still exist, and it also ensures that the data is available in case it is needed for a support case.

See also "Managing long-running transactions".

Making logs available in a RAC configuration

In a RAC configuration, Extract must have access to the online and archived logs for all nodes in the cluster, including the one where GoldenGate is installed.

Specifying the archive configuration

If the archived logs reside in a location other than the Oracle default, specify that location with the ALTARCHIVELOGDEST option of the TRANLOGOPTIONS parameter in the Extract parameter file.

You might also need to use the ALTARCHIVEDLOGFORMAT option of TRANLOGOPTIONS if the format that is specified with the Oracle parameter LOG_ARCHIVE_FORMAT contains sub-directories. ALTARCHIVEDLOGFORMAT specifies an alternate format that removes the sub-directory from the path. For example, %T/log_%t_%s_%r.arc would be changed to log_%t_%s_%r.arc. As an alternative to using ALTARCHIVEDLOGFORMAT, you can create the sub-directory manually, and then move the log files to it.

Configuring GoldenGate to read archived logs only

You can configure the Extract process to read exclusively from the archived logs. This is known as *Archived Log Only* (ALO) mode. In this mode, Extract only reads from archived logs that are stored in a specified location. This mode allows GoldenGate to use the archived redo logs from a standby database or any other database as the data source for GoldenGate. The online logs will not be used.

Limitations of ALO mode

- ALO mode is supported for Oracle9i and later.
- Log resets (RESETLOG) cannot be done on the source database after the standby database is created.
- The extraction and replication of DDL are not supported in ALO mode.
- ALO cannot be used on a standby database if the production system is RAC and the standby database is non-RAC. ALO mode is supported on the production system.
- ALO on RAC requires a dedicated connection to the source server. If that connection is lost, GoldenGate processing will stop.
- Supplemental logging at the table level and the database level must be enabled for the tables from the source database.
- ALO mode does not support archive log files in ASM mode. The archive log files must be outside the ASM environment for us to read them.

Configuring Extract for ALO mode

1. To connect to the database when GoldenGate is running on a different server, use a SQL*Net connect string in the USERID parameter. This connect string can then be used for logins for all GoldenGate processes and from GGSCI. Make certain that SQL*Net is configured properly to connect to a remote server, including the correct entries in a TNSNAMES file. Example USERID statement:

   ```
   USERID ggext@ora10g01, PASSWORD ggs123
   ```

NOTE If you have a standby server that is local to the server that GoldenGate is running on, you do not need to use a connect string in USERID. You can just supply the user login name.

2. Use the Extract parameter TRANLOGOPTIONS with the ARCHIVEDLOGONLY option. This option forces Extract to operate in ALO mode against a primary or logical standby database, as determined by a value of PRIMARY or LOGICAL STANDBY in the db_role column of the v$database view. The default on a primary or logical standby database is to read the online logs. This parameter is not needed if using ALO mode on a physical standby database. Extract automatically operates in ALO mode if it detects that the database has a role of PHYSICAL STANDBY.
3. Add the Extract group by using a timestamp or by using the SEQNO and RBA options of the ADD EXTRACT command. It is best to give Extract a known start point at which to begin extracting data, rather than by using the NOW argument. The start time of "NOW" corresponds to the time of the current *online* redo log, but an ALO Extract cannot read the online logs, so it must wait for that log to be archived when Oracle switches logs. The timing of the switch depends on the size of the redo logs and the volume of database activity, so there might be a lag between when you start the Extract and when data starts being captured. This can happen in both regular and RAC database configurations.

   NOTE If Extract appears to stall while operating in ALO mode, see the *GoldenGate for Windows and UNIX Troubleshooting and Performance Tuning Guide* for help with diagnosing the problem.

Managing the effects of row chaining

When rows are fragmented, GoldenGate does extra work, and so do the database processes. To avoid this problem, monitor the fragmentation levels and then consider doing the following on a routine basis:

- Remove fragmented tables from the GoldenGate configuration, reorganize them, and resynchronize them. Then add them back. Do not start the reorganization while Extract is capturing data from the tables.
- Instead of reorganizing a table, you could copy the rows from it, then delete and reinsert them again. This alters report statistics, however.
- If you cannot manage the row fragmentation in those ways, you can process heavily fragmented tables through a separate Extract group. To configure multiple process groups, see the *GoldenGate for Windows and UNIX Administrator Guide*.

Adjusting cursors

The Extract process maintains cursors for queries that fetch data and also for SQLEXEC operations. Without enough cursors, Extract must age more statements. By default, Extract maintains as many cursors as allowed by the Extract MAXFETCHSTATEMENTS parameter. You might find that the value of this parameter needs to be increased. If so, you might also need to adjust the maximum number of open cursors that are permitted by the database.

Managing long-running transactions

Check your applications for how long their transactions remain open. Extract does not write open transactions to the trail until the applications commit them. Until the commits are received, Extract keeps track of these transactions and maintains a checkpoint in the log that contains the start point of the oldest one. If Extract stops, it reads that checkpoint when it starts again, and then it searches for that log. If the log is not there, Extract returns an error. This situation is likely to occur in a high-volume setting, where the logs age quickly and must be removed from the system frequently to conserve disk space.

To manage long-running transactions

Use the WARNLONGTRANS parameter to specify a length of time that a transaction can be open before Extract generates a warning message that the transaction is long-running. Also use WARNLONGTRANS to control the frequency with which GoldenGate checks for long-running transactions.

Use the SHOWTRANS, SKIPTRANS, and FORCETRANS options of the SEND EXTRACT command to view open transactions, to remove any transaction from the Extract memory structure, or to force any transaction to be written to the trail as a committed transaction.

NOTE Command output may vary somewhat from the examples shown due ongoing enhancements of the GoldenGate software.

Figure 1 SHOWTRANS default output

```
Oldest redo log file necessary to restart Extract is:
Redo Thread 1, Redo Log Sequence Number 148, SCN 30816254, RBA 17319664
------------------------------------------------------------
XID                : 5.15.52582
Items              : 30000
Extract            : JC108XT
Redo Thread        : 1
Start Time         : 2006-05-18:12:51:27
SCN                : 20634955
Redo Seq           : 103
Redo RBA           : 18616848
Status             : Running
------------------------------------------------------------
XID                : 7.14.48657
Items              : 30000
Extract            : JC108XT
Redo Thread        : 1
Start Time         : 2006-05-18:12:52:14
SCN                : 20635145
Redo Seq           : 103
Redo RBA           : 26499088
Status             : Running
```

Figure 2 SHOWTRANS output with TABULAR in effect (view is truncated on right)

```
XID         Items  Extract   Redo Thread  Start Time
5.15.52582  30000  JC108XT1            2006-05-18:12:52:14
```

Figure 3 SHOWTRANS FILE <name> DETAIL

```
Dumping transaction memory at 2006-07-21 13:36:54.
Record #1:
Header (140 bytes):
       0: 0000 0A4A 0000 FFFF 0000 0000 0057 6C10        ...J.........Wl.
      16: 02FF 3F50 FF38 7C40 0303 4141 414E 5A77        ..?P.8|@..AAANZw
      32: 4141 4641 4141 4B6F 4941 4144 0041 4141        AAFAAAKoIAAD.AAA
      48: 4E5A 7741 4146 4141 414B 6F49 4141 4400        NZwAAFAAAKoIAAD.
      64: 4141 414E 5A77 414A 2F41 4142 7A31 7741        AAANZwAJ/AABz1wA
      80: 4141 0041 4141 4141 4141 4141 4141 4141        AA.AAAAAAAAAAAA
      96: 4141 4141 4100 0000 0140 FF08 0003 0000        AAAAA....@......
     112: 0000 0000 0000 70FF 0108 FFFF 0001 4A53        ......p.......JS
     128: 554E 2E54 4355 5354 4D45 5200                    UN.TCUSTMER.

Data (93 bytes):
       0: 2C00 0400 0400 0000 0100 0200 0300 0000        ,...............
      16: 0000 0000 0800 0000 1800 0000 2000 0400        ............ ...
      32: 1000 0600 0200 0000 284A 414E 456C 6C6F        ........(JANEllo
      48: 6352 4F43 4B59 2046 4C59 4552 2049 4E43        cROCKY FLYER INC
      64: 2E44 454E 5645 5220 6E43 4F20 7365 7400        .DENVER nCO set.
      80: 0000 0000 0000 0C00 0000 0000 00                ..............
```

When analyzing the summary output of SHOWTRANS, understand that it shows all currently running transactions on the database (as many as will fit into a predefined buffer). Extract must track every open transaction, not just those that contain operations on tables configured for GoldenGate, because it is not known whether operations on configured tables will be added to a transaction at some point in the future.

The Items field of the SHOWTRANS output shows the number of operations in the transaction that have been captured by GoldenGate so far, not the total number of operations in the transaction. If none of the operations are for configured tables, or if only some of them are, then Items could be 0 or any value less than the total number of operations.

The Start Time field shows the timestamp of the first operation that GoldenGate extracts from a transaction, not the actual start time of the transaction itself.

Setting fetch options

To process certain update records from the redo log, GoldenGate fetches additional row data from the source database. The way the fetch is performed depends on the version of the Oracle database and might require setting some parameters.

To determine which operations require a fetch

- Oracle 8.x or earlier: GoldenGate fetches row values directly from the database if a redo record that it is processing affects a row that is chained or contains a LOB. The fetch, however, reflects the current state of the row, which might be newer (updated or deleted) than the image at the point in time reflected in the redo record.

- Oracle 9i or later: GoldenGate fetches data for operations that contain LOBs, user-defined types, nested tables, and XMLType. By default, GoldenGate uses Flashback Query to fetch the values from the undo (rollback) tablespaces. That way, GoldenGate can reconstruct a read-consistent row image as of a specific time or SCN to match the redo record.

To configure the database for best fetch results

For best fetch results, configure the source database as follows:

1. Set a sufficient amount of redo retention by setting the Oracle initialization parameters UNDO_MANAGEMENT and UNDO_RETENTION as follows (in seconds).

   ```
   UNDO_MANAGEMENT=AUTO
   UNDO_RETENTION=86400
   ```

   UNDO_RETENTION can be adjusted upward in high-volume environments.

2. Calculate the space required in the undo tablespace by using the following formula.

   ```
   <undo space> = <UNDO_RETENTION> * <UPS> + <overhead>
   ```

   Where:

   - <undo space> is the number of undo blocks.
   - <UNDO_RETENTION> is the value of the UNDO_RETENTION parameter (in seconds).
   - <UPS> is the number of undo blocks for each second.
   - <overhead> is the minimal overhead for metadata (transaction tables, etc.).

   Use the system view V$UNDOSTAT to estimate <UPS> and <overhead>.

3. For tables that contain LOBs, do one of the following:

   - Set the LOB storage clause to RETENTION. This is the default for tables created when UNDO_MANAGEMENT is set to AUTO.
   - If using PCTVERSION instead of RETENTION, set PCTVERSION to an initial value of 25. You can adjust it based on the fetch statistics reported with the STATS EXTRACT command (see "To configure GoldenGate fetch options"). If the value of the STAT_OPER_ROWFETCH CURRENTBYROWID or STAT_OPER_ROWFETCH_CURRENTBYKEY field in these statistics is high, increase PCTVERSION in increments of 10 until the statistics show low values.

4. Grant the following privileges to the GoldenGate Extract user:

   ```
   GRANT FLASHBACK ANY TABLE TO <db_user>
   ```

   Or ...

   ```
   GRANT FLASHBACK ON <owner.table> TO <db_user>
   ```

To configure GoldenGate fetch options

GoldenGate provides the following parameters to manage fetching.

- To view fetch statistics on-demand, use the STATS EXTRACT command with the REPORTFETCH option.

- To set the STATS EXTRACT command so that it always shows fetch statistics, use the Extract parameter STATOPTIONS with the REPORTFETCH option.
- To control the number of open cursors for prepared queries that Extract maintains in the source database, use the Extract parameter MAXFETCHSTATEMENTS.
- To control the default fetch behavior of Extract for an Oracle 9i database, use the FETCHOPTIONS parameter with the USESNAPSHOT or NOUSESNAPSHOT option. They control whether Extract performs a flashback query or fetches the current image from the table.
- To handle failures of flashback queries, use the FETCHOPTIONS parameter with the USELATESTVERSION or NOUSELATESTVERSION option. A flashback query can fail if the undo retention has expired or the structure of a table has changed. These options control whether Extract fetches the current image from the table or ignores the failure.
- To control the response by Replicat when it processes trail records that include fetched data or column-missing conditions, use the Replicat parameter REPFETCHEDCOLOPTIONS.

Preparing tables for processing

The following table attributes must be addressed in a GoldenGate environment.

Disabling triggers and cascade delete constraints

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

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

Assigning row identifiers

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

How GoldenGate determines the kind of row identifier to use

GoldenGate selects a row identifier to use in the following order of priority:

1. Primary key

2. First unique key alphanumerically with no virtual columns, no UDTs, no function-based columns, and no nullable columns
3. First unique key alphanumerically with no virtual columns, no UDTs, or no function-based columns, but can include nullable columns
4. If none of the preceding key types exist (even though there might be other types of keys defined on the table) GoldenGate constructs a pseudo key of all columns that the database allows to be used in a unique key, excluding virtual columns, UDTs, function-based columns, and any columns that are explicitly excluded from the GoldenGate configuration.

NOTE If there are other, non-usable keys on a table (such as one that includes a virtual column), or if there are no keys at all on the table, GoldenGate logs an appropriate message to the report file. Constructing a key of all of the columns impedes the performance of GoldenGate on the source system. On the target, this key causes Replicat to use a larger, less efficient WHERE clause.

How to specify your own key for GoldenGate to use

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

Getting the database to log key values

Use the ADD TRANDATA command in GGSCI to configure the database to log the key values whenever it logs a row change, so that they are available to GoldenGate in the redo record. By default, the database only logs column values that are changed.

ADD TRANDATA will:

- Force the logging of primary or unique key columns, or all of the columns to use as a key if there are no defined keys (except non-supported column types). In Oracle 8i, ADD TRANDATA installs an update trigger that sets the values to themselves, causing them to be logged. In Oracle 9i and later, ADD TRANDATA creates a supplemental log group that contains the required columns.

  Or...

- Log non-key columns that will be used in a KEYCOLS clause. This prevents ADD TRANDATA from logging all of the columns of the table when it determines there is no primary or unique key; the KEYCOLS specification in the parameter file is not checked until processing starts.

ADD TRANDATA must be performed before you start GoldenGate processing. Before using ADD TRANDATA, see the *GoldenGate for Windows and UNIX Reference Guide* for additional instructions and options.

To capture key values with ADD TRANDATA

1. On the source system, run GGSCI from the GoldenGate directory.

2. In GGSCI, issue the following command to log on to the database.

   ```
   DBLOGIN USERID <user>, PASSWORD <password>
   ```

   Where: <user> is a database user who has privilege to create triggers or enable table-level supplemental logging, and <password> is that user's password.

3. Issue the ADD TRANDATA command.

   ```
   ADD TRANDATA <table> [, COLS <columns>] [, NOKEY] [, USETRIGGER]
   ```

   Where:

   - <table> is the owner and name of the table. You can use a wildcard for the table name but not the owner name.
   - COLS <columns> logs non-key columns that are specified with KEYCOLS.
   - NOKEY prevents the logging of the primary key or unique key. Requires using a KEYCOLS clause in TABLE or MAP and logging the KEYCOLS columns with COL.
   - USETRIGGER forces GoldenGate to install an update trigger instead of a supplemental log group. Required only if 8i compatibility is enabled on a 9i or later database.

4. (Oracle 9i and later) Log in to SQL*Plus as a user with ALTER SYSTEM privilege, and issue the following command to enable minimal supplemental logging at the database level. This is required to process updates to primary keys and chained rows.

   ```
   ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
   ```

5. To start the supplemental logging, switch the log files.

   ```
   ALTER SYSTEM SWITCH LOGFILE;
   ```

6. Verify that supplemental logging is enabled at the database level with this command:

   ```
   SELECT SUPPLEMENTAL_LOG_DATA_MIN FROM V$DATABASE;
   ```

   - For Oracle 9i, the output of the query must be YES.
   - For Oracle 10g, the output of the query must be YES or IMPLICIT.

7. (Oracle 8i and 8i compatibility) Use the TRANLOGOPTIONS parameter with the FETCHCHAINEDUPDATES option in the Extract parameter file.

8. If using the COLS option to log KEYCOLS columns, create a unique index for these columns on the target, to optimize row retrieval.

Limiting row changes in tables that do not have a key

If a target table has no primary key or unique key, duplicate rows can exist. It is possible for GoldenGate to update or delete too many rows in the target table, causing the source and target data to go out of synchronization without error messages to alert you. To limit the number of rows that are updated, use the DBOPTIONS parameter with the LIMITROWS option in the Replicat parameter file. LIMITROWS can increase the performance of GoldenGate on the target system because only one row is processed.

Deferring constraint checking

Constraints might need to be modified in the following cases:

Source and target not both set to DEFERRED

If constraints are DEFERRABLE on the source, the constraints on the target must also be DEFERRABLE. As an alternative, you can use the following parameter statement at the root level of the Replicat parameter file to defer the constraints only for Replicat sessions.

```
SQLEXEC ("alter session set constraint deferred")
```

Key updates affect multiple rows

Replicat might need to set constraints to DEFERRED if it is possible that an update transaction could affect the primary keys of multiple rows. Called a *transient primary key update* within GoldenGate, this kind of operation typically uses an *x+n* formula or other form of manipulation that shifts the values and causes a new value to be the same as an old one.

The following illustrates a sequence of value changes that can cause this condition if constraints are not deferred. The example assumes the primary key column is "CODE" and the current key values (before the updates) are 1, 2, and 3.

```
update item set code = 2 where code = 1;
update item set code = 3 where code = 2;
update item set code = 4 where code = 3;
```

In this example, when Replicat applies the first update to the target, there is an error because the key value of 2 already exists in the table. The Replicat transaction returns constraint violation errors. By default, Replicat does not handle these violations and abends.

To enable Replicat to manage these updates:

- Create the constraints as DEFERRABLE on the target tables. Either INITIALLY DEFERRED or INITIALLY IMMEDIATE can be specified.
- Use the Replicat parameter HANDLETPKUPDATE to enable Replicat to set constraints as INITIALLY DEFERRED in its transactions (check constraints when the transaction is committed).

If constraints are not DEFERRABLE, Replicat handles the errors according to existing rules specified with the HANDLECOLLISIONS and REPERROR parameters, or else it abends.

Replicating Oracle TDE data

If any tables have columns that use Transparent Data Encryption (TDE), check them against the following GoldenGate limitations:

- The table that contains the TDE columns must have a primary or unique key.
- Columns that use TDE cannot be part of the primary key.

To enable processing of TDE-protected data

TDE columns are encrypted in the data files and in the redo log, so Extract must be configured to fetch the clear-text values from the database. To trigger this fetch, use the

FETCHCOLS and FETCHMODCOLS[EXCEPT] options of the Extract TABLE parameter. FETCHCOLS forces a fetch of values that are not in the redo log, and FETCHMODCOLS or FETCHMODCOLS[EXCEPT] forces a fetch of values that are in the logs. Used together, these parameters ensure that the TDE columns are always fetched from the database.

The following is an example of how to configure Extract to support TDE. In this example, the TDE column is credit_card_number.

```
EXTRACT ext_tde
USERID ggs, password ggs
RMTHOST sysb, mgrport 4261
RMTTRAIL C:\ggs\oracle\v8100\dirdat\td
TABLE ab.payment_info, FETCHCOLS (credit_card_number), &
FETCHMODCOLS (credit_card_number);
```

Ensuring correct handling of Oracle Spatial objects

If you are replicating georaster tables (tables that contain one or more columns of SDO_GEORASTER object type), follow these instructions to configure GoldenGate to capture and replicate them properly.

Mapping the tables

You must create a TABLE statement and a MAP statement for the georaster tables and also for the related raster data tables.

Sizing the XML memory buffer

You might need to change the size of the memory buffer that stores the embedded SYS.XMLTYPE attribute of the SDO_GEORASTER data type. This buffer is controlled by the DBOPTIONS parameter with the XMLBUFSIZE option. The default is 1048576 bytes (1MB) and the maximum is 10 MB. If the data exceeds the buffer size, Extract will abend.

Evaluate your Spatial data before starting GoldenGate processes, so that you can adjust the buffer to suit the size of the XML data. If the METADATA attribute of the SDO_GEORASTER data type in any of the values exceeds the default of 1 MB, you will have to increase this buffer.

For more information about DBOPTIONS, see the *GoldenGate for Windows and UNIX Reference Guide*.

Handling triggers on the georaster tables

Every georaster table has a trigger associated with it to populate the raster data table.

- This trigger must always be enabled in both the source and target environments to ensure consistency of the spatial data.
- When a row in the georaster table is deleted, the trigger cascades the delete to the raster data table for certain rows. In a manner that is similar to cascaded deletes, the order of the arrival of the parent and child deletes will cause ORA-01403 No data found errors unless the error is handled programmatically through Replicat. (For more information, see “Disabling triggers and cascade delete constraints” on page 48.)

To handle the cascaded deletes, you can use the REPERROR option of the MAP parameter statement that you create for each raster data table. Use Oracle error 1403 as the SQL error, and use any of the response options as the error handling.

Example A sufficient way to handle the errors is simply to use REPERROR with DISCARD to discard the cascaded delete that triggers them. The trigger on the target georaster table performs the delete to the raster data table, so the replicated one is not needed.

```
MAP geo.st_rdt, TARGET geo.st_rdt, REPERROR (-1403, DISCARD) ;
```

Example If you need to keep an audit trail of the error handling, a more comprehensive method is to use REPERROR with EXCEPTION to invoke exceptions handling. For this, you create an exceptions table and map the source raster data table twice: once to the actual target raster data table (with REPERROR handling the 1403 errors), and again to the exceptions table, which will capture the 1403 error and other relevant information by means of a COLMAP clause. When using exceptions handling like this, you must use the ALLOWDUPTARGETMAP parameter to keep Replicat from abending on the dual source mapping.

This example provides a Replicat parameter file that contains the required parameters, and it provides a sample script that creates an exceptions table. Note that a macro is used in the parameter file to populate the TARGET and COLMAP portions of the exceptions MAP statements. The required INSERTALLRECORDS and EXCEPTIONSONLY parameters are also included in the macro. The macro prevents having to type the same information over again for each of the MAP statements.

Replicat parameter file

```
REPLICAT rgeoras
SETENV (ORACLE_SID=tgt111)
USERID geo, PASSWORD xxxxx, ENCRYPTKEY DEFAULT

ASSUMETARGETDEFS
DISCARDFILE ./dirrpt/rgeoras.dsc, purge
ALLOWDUPTARGETMAP

-- This starts the macro
MACRO #exception_handling
BEGIN
, TARGET geo.exceptions
, COLMAP ( rep_id = "1"
, table_name = @GETENV ("GGHEADER", "TABLENAME")
, errno = @GETENV ("LASTERR", "DBERRNUM")
, dberrmsg = @GETENV ("LASTERR", "DBERRMSG")
, optype = @GETENV ("LASTERR", "OPTYPE")
, errtype = @GETENV ("LASTERR", "ERRTYPE")
, logrba = @GETENV ("GGHEADER", "LOGRBA")
, logposition = @GETENV ("GGHEADER", "LOGPOSITION")
, committimestamp = @GETENV ("GGHEADER", "COMMITTIMESTAMP")
)
, INSERTALLRECORDS
, EXCEPTIONSONLY ;
END;
-- This ends the macro
```

```
EXTTRAIL ./dirdat/eg

-- Mapping of regular and georaster tables. Requires no exception handling.
-- Replicat abends on errors, which is its default error handling.
MAP geo.blob_table, TARGET geo.blob_table ;
MAP geo.georaster_table, TARGET geo.georaster_table ;
MAP geo.georaster_table2, TARGET geo.georaster_table2 ;
MAP geo.georaster_tab1, TARGET geo.georaster_tab1 ;
MAP geo.georaster_tab2, TARGET geo.georaster_tab2 ;
MAP geo.mv_georaster_table1, TARGET geo.mv_georaster_table1 ;

-- Mapping of raster data tables. Requires exception handling for 1403 errors.
MAP geo.st_rdt_3_table, TARGET geo.st_rdt_3_table, REPERROR (-1403, EXCEPTION)
;
MAP geo.st_rdt_3_table #exception_handling()
MAP geo.rdt_1_table, TARGET geo.rdt_1_table, REPERROR (-1403, EXCEPTION) ;
MAP geo.rdt_1_table #exception_handling()
MAP geo.rdt_2_table, TARGET geo.rdt_2_table, REPERROR (-1403, EXCEPTION) ;
MAP geo.rdt_2_table #exception_handling()
MAP geo.mv_rdt_1_table, TARGET geo.mv_rdt_1_table, REPERROR (-1403, EXCEPTION)
;
MAP geo.mv_rdt_1_table #exception_handling()
```

Sample script that creates an exceptions table

```
drop table exceptions
/

create table exceptions
( rep_id number
, table_name varchar2(61)
, errno number
, dberrmsg varchar2(4000)
, optype varchar2(20)
, errtype varchar2(20)
, logrba number
, logposition number
, committimestamp timestamp
)
/
```

NOTE When using an exceptions table for numerous tables, someone should monitor its growth.

For more information about using an exceptions table and using macros, see the *GoldenGate for Windows and UNIX Administrator Guide*.

For more information about MAP options, see the *GoldenGate for Windows and UNIX Reference Guide*.

Consolidating fetching

Oracle creates multiple updates for the same row when updating an out-of-row LOB, each of which triggers a fetch to the database by Extract to get the data. To avoid multiple fetches and to prevent raster captures from expanding the trail files, you can configure Extract to combine the series of operations into one operation that reflects the net result. Only a single fetch would then be required for that single operation. Use the Extract parameter FETCHOPTIONS with the undocumented options of SUPPRESSDUPLICATES to consolidate the operations that require fetching.

Managing LOB caching

Because Replicat must write LOB data to the target database in fragments, there is the potential for excessive I/O between Replicat and the database. To minimize the effect of I/O on the system, Replicat caches LOB fragments in a buffer of a specified size and only performs a write when the buffer is full. The default LOB write size is 32k.

For example, if the buffer is 25,000 bytes in size, then that is the size of the block that Replicat writes to the database. Given a LOB of 100,000 bytes, Replicat generates I/O only four times to process this LOB.

- To control the size of the LOB buffer, use the Replicat parameter DBOPTIONS with the LOBWRITESIZE <size> option. The higher the value, the fewer I/O calls that are made by Replicat to the database server to write the whole LOB to the database.
- To disable Oracle's LOB caching, use the DISABLELOBCACHING option of DBOPTIONS. By default, Replicat enables Oracle's LOB caching mechanism. If Oracle's LOB caching is disabled, whatever is sent by Replicat to Oracle in one I/O call is written directly to the database media. If Oracle's caching is enabled, whatever is sent by Replicat to Oracle in one I/O call will most likely be cached in Oracle's internal LOB buffer before being applied to the database media.

Additional requirements and procedures for Oracle RAC

General requirements

- All nodes in the RAC cluster must have synchronized system clocks. These clocks also must be synchronized with the clock on the system where Extract is executed. GoldenGate compares the local system's time to commit timestamps to make critical decisions. For information about synchronizing system clocks, consult www.ntp.org or your systems administrator. See also the IOLATENCY option of the THREADOPTIONS parameter in the *GoldenGate for Windows and UNIX Reference Guide*.
- All nodes in the cluster must have the same COMPATIBLE parameter setting.

GoldenGate parameters settings for RAC

1. On AIX and Solaris machines, use the Extract parameter THREADOPTIONS with the BINDCPU <n> option. This parameter handles thread-safety issues related to memory that is updated by different processors.

2. GoldenGate queues data in memory before sending it to the target system. The INQUEUESIZE and OUTQUEUESIZE options of the THREADOPTIONS parameter determine how much data to queue. If needed, you can increase the performance of Extract on Oracle RAC by tuning these parameters.
3. GoldenGate detects orphaned transactions, which can occur when a node fails during a transaction and Extract cannot capture the rollback. Although the database performs the rollback on the failover node, the transaction would otherwise remain in GoldenGate's transaction list indefinitely and prevent further checkpointing for the Extract thread that was processing the transaction. By default, GoldenGate purges these transactions from its list after they are confirmed as orphaned. To control this behavior, use the TRANLOGOPTIONS parameter with the PURGEORPHANEDTRANSACTIONS | NOPURGEORPHANEDTRANSACTIONS and TRANSCLEANUPFREQUENCY options. This functionality can be controlled on demand with the SEND EXTRACT command in GGSCI.

Special procedures on RAC

1. If the primary database instance against which GoldenGate is running stops or fails for any reason, Extract will abend. To resume processing, you can restart the instance, or you can mount the GoldenGate binaries to another node where the database is running and then restart the GoldenGate processes. Stop the Manager process on the original node before starting GoldenGate processes from another node.
2. Any time the number of redo threads changes, the Extract group must be dropped and re-created. For the recommended procedure, see the *GoldenGate for Windows and UNIX Administrator Guide*.
3. To write SQL operations to the trail, Extract must make sure that there are no other operations from other RAC nodes that precede those in the current redo log that it is reading. For example, if a log contains operations that were performed from 1:00 a.m. to 2:00 a.m., and the log from Node 2 contains operations that were performed from 1:30 a.m. to 2:30 a.m., then only those operations up to, and including, the 2:00 a.m. one can be moved to the server where the main Extract is coordinating the redo data. Extract must ensure that there are no more operations between 2:00 a.m. and 2:30 a.m. that need to be captured.

   In active-passive environments, that requirement means that you might need to perform some operations and archive log switching on the passive node, to ensure that operations from the active node are passed to the passive node. This eliminates any issues that could arise from a slow archiver process, failed network links, and other latency issues caused by moving archive logs from the Oracle nodes to the server where the main Extract is coordinating the redo data.
4. To process the last transaction in a RAC cluster before shutting down Extract, insert a dummy record into a source table that GoldenGate is replicating, and then switch log files on all nodes. This will update the Extract checkpoint and confirm to the process that all available archive logs are available to read. It also confirms that all transactions in those archive logs are captured and written to the trail in the correct order.

CHAPTER 5

Managing the Oracle DDL replication environment

This chapter contains instructions for making changes to the database environment or the GoldenGate environment when the GoldenGate DDL objects that you installed in Chapter 3 still exist on the system.

For instructions on configuring GoldenGate DDL support, see the *GoldenGate for Windows and UNIX Administrator Guide*.

Enabling and disabling the DDL trigger

You can enable and disable the trigger that captures DDL operations without making any configuration changes within GoldenGate. The following scripts control the DDL trigger.

- ddl_disable: Disables the trigger. No further DDL operations are captured or replicated after you disable the trigger.
- ddl_enable: Enables the trigger. When you enable the trigger, GoldenGate starts capturing current DDL changes, but does not capture DDL that was generated while the trigger was disabled.

Before running these scripts, disable all sessions that ever issued DDL. Otherwise the database might generate an ORA-04021 error. Do not use these scripts if you intend to maintain consistent DDL on the source and target systems.

Maintaining the DDL marker table

You can purge rows from the marker table at any time. It does not keep DDL history. To purge the marker table, use the Manager parameter PURGEMARKERHISTORY. Manager gets the name of the marker table from one of the following:

1. The name given with the MARKERTABLE <table> parameter in the GLOBALS file, if specified.
2. The default name of GGS_MARKER.

Instead of using PURGEMARKERHISTORY to purge the marker table, you can use the Manager parameter PURGEOLDHISTORY. Both parameters provide options to specify maximum and minimum lengths of time to keep a row, based on the last modification date. For more information, see the *GoldenGate for Windows and UNIX Reference Guide*.

Deleting the DDL marker table

Do not delete the DDL marker table unless you want to discontinue synchronizing DDL. The marker table and the DDL trigger are interdependent. An attempt to drop the marker table will fail if the DDL trigger is enabled. This is a safety measure to prevent the trigger from becoming invalid and missing DDL operations. If you remove the marker table, the following error will be generated:

```
"ORA-04098: trigger 'SYS.GGS_DDL_TRIGGER_BEFORE' is invalid and failed
re-validation"
```

The proper way to remove a GoldenGate DDL object depends on your plans for the rest of the DDL environment. To choose the correct procedure, see one of the following:

- "Changing DDL object names after installation" on page 60
- "Restoring an existing DDL environment to a clean state" on page 62
- "Removing the DDL objects from the system" on page 63

Maintaining the DDL history table

You can purge the DDL history table to control its size, but this must be done carefully. The DDL history table maintains the integrity of the DDL synchronization environment. Purges to this table cannot be recovered through the GoldenGate interface.

To maintain the DDL history table

1. To prevent any possibility of DDL history loss, make regular full backups of the history table.
2. To ensure recoverability of purged DDL, enable Oracle Flashback for the history table. Set the flashback retention time well past the point where it could be needed. For example, if your full backups are at most one week old, retain two weeks of flashback. GoldenGate can be positioned backward into the flashback for reprocessing.
3. As an additional safeguard, consider including the DDL history table in your GoldenGate DML replication configuration. Note: Any passwords that are stored in this table will be replicated to the target system.
4. If possible, purge the DDL history table manually. This will ensure that essential rows are not purged accidentally. If you require an automated purging mechanism, use the PURGEDDLHISTORY parameter in the Manager parameter file. You can specify maximum and minimum lengths of time to keep a row. For more information, see the *GoldenGate for Windows and UNIX Reference Guide*.

NOTE Temporary tables created by GoldenGate to increase performance might be purged at the same time as the DDL history table, according to the same rules. The names of these tables are derived from the name of the history table, and their purging is reported in the Manager report file. This is normal behavior.

Deleting the DDL history table

Do not delete the DDL history table unless you want to discontinue synchronizing DDL. The history table contains a record of DDL operations that were issued. If you delete this table prematurely, you could compromise the integrity of the target DDL.

The history table and the DDL trigger are interdependent. An attempt to drop the history table will fail if the DDL trigger is enabled. This is a safety measure to prevent the trigger from becoming invalid and missing DDL operations. If you remove the history table, the following error will be generated:

```
"ORA-04098: trigger 'SYS.GGS_DDL_TRIGGER_BEFORE' is invalid and failed
re-validation"
```

The proper way to remove a GoldenGate DDL object depends on your plans for the rest of the DDL environment. To choose the correct procedure, see one of the following:

- "Changing DDL object names after installation" on page 60
- "Restoring an existing DDL environment to a clean state" on page 62
- "Removing the DDL objects from the system" on page 63

Purging the DDL trace file

To prevent the DDL trace file from consuming excessive disk space, run the ddl_cleartrace script on a regular basis. This script deletes the file, but GoldenGate will create it again.

The default name of the DDL trace file is ggs_ddl_trace.log. It is in the USER_DUMP_DEST directory of Oracle. The ddl_cleartrace script is in the GoldenGate directory.

Applying database patches and upgrades when DDL support is enabled

Database patches and upgrades usually invalidate the GoldenGate DDL trigger and other GoldenGate DDL objects. Before applying a database patch, do the following.

1. Disable the GoldenGate DDL trigger by running the following script.

   ```
   @ddl_disable
   ```

2. Apply the patch.
3. Enable the DDL trigger by running the following script.

   ```
   @ddl_enable
   ```

NOTE Database upgrades and patches generally operate on Oracle objects. Because GoldenGate filters out those objects automatically, DDL from those procedures is not replicated when replication starts again.

To avoid recompile errors after the patch or upgrade, which are caused if the trigger is not disabled before the procedure, consider adding calls to @ddl_disable and @ddl_enable at the appropriate locations within your scripts.

Applying GoldenGate patches and upgrades when DDL support is enabled

Follow these steps to apply a patch or upgrade to the DDL objects. This procedure preserves the current DDL synchronization configuration.

NOTE Do not use this procedure for an upgrade to this version of GoldenGate from a GoldenGate version that does not support DDL statements that are larger than 2

MB (pre-version 10.4). To upgrade in this case, follow the instructions in "Restoring an existing DDL environment to a clean state" on page 62.

1. Run GGSCI. Keep the session open for the duration of this procedure.
2. Stop the Extract process to stop DDL capture.

   ```
   STOP EXTRACT <group>
   ```

3. Stop the Replicat process to stop DDL replication.

   ```
   STOP REPLICAT <group>
   ```

4. Download or extract the patch or upgrade files according to the instructions provided by GoldenGate.
5. Change directories to the GoldenGate installation directory.
6. Run SQL*Plus and log in as a user that has SYSDBA privileges.
7. Disable all sessions that ever issued DDL. Otherwise the database might generate an ORA-04021 error.
8. Run the ddl_disable script to disable the DDL trigger.
9. Run the ddl_setup script. You will be prompted for:
   - The name of the GoldenGate schema. If you changed the schema name, use the new one.
   - The installation mode: Select the NORMAL mode. This mode recompiles the DDL environment without removing DDL history.
10. Run the ddl_enable.sql script to enable the DDL trigger.
11. In GGSCI, start Extract to resume DDL capture.

    ```
    START EXTRACT <group>
    ```

12. Start the Replicat process to start DDL replication.

    ```
    START REPLICAT <group>
    ```

Changing DDL object names after installation

Follow these steps to change the names of the GoldenGate DDL schema or other DDL objects after they are installed. This procedure preserves the continuity of source and target DDL operations.

1. Run GGSCI. Keep the session open for the duration of this procedure.
2. Stop Extract to stop DDL capture.

   ```
   STOP EXTRACT <group>
   ```

3. Stop the Replicat process to stop DDL replication.

   ```
   STOP REPLICAT <group>
   ```

4. Change directories to the GoldenGate installation directory.

5. Run SQL*Plus and log in as a user that has SYSDBA privileges.
6. Disable all sessions that ever issued DDL. Otherwise the database might generate an ORA-04021 error.
7. Run the ddl_disable script to disable the DDL trigger.
8. To change the DDL schema name, specify the new name in the local GLOBALS file.

   GGSCHEMA <new_schema_name>

9. To change the names of any other objects, do the following:
   - Specify the new names in the params.sql script. Do not run this script.

   and...
   - If changing objects in Table 9, specify the new names in the local GLOBALS file. The correct parameters to use are listed in the Parameter column of this table.

Table 9 GLOBALS parameters for changing DDL object names

Object	Parameter
Marker table	MARKERTABLE <new_table_name>[1]
History table	DDLTABLE <new_table_name>

[1] Do not qualify the name of any of these tables. The schema name for these table must be either the user specified with GGSCHEMA or the current user, if GGSCHEMA is not specified in GLOBALS.

10. If using a new schema for the DDL synchronization objects, create it now.
11. Change directories to the GoldenGate installation directory.
12. Run SQL*Plus and log in as a user that has SYSDBA privileges.
13. Run the ddl_setup script. You will be prompted for:
    - The name of the GoldenGate schema. If you changed the schema name, use the new one.
    - The installation mode: Select the NORMAL mode to recompile the DDL environment without removing the DDL history table.
14. Run the ddl_enable.sql script to enable the DDL trigger.
15. In GGSCI, start Extract to resume DDL capture.

    ```
    START EXTRACT <group>
    ```

16. Start the Replicat process to start DDL replication.

    ```
    START REPLICAT <group>
    ```

Restoring an existing DDL environment to a clean state

Follow these steps to completely remove, and then reinstall, the GoldenGate DDL objects. This procedure creates a new DDL environment, but removes DDL history.

> **NOTE** Due to object interdependencies, all objects must be removed and reinstalled in this procedure.

1. If you are performing this procedure in conjunction with the installation of a new GoldenGate version, download and install the GoldenGate files, and create or update process groups and parameter files as necessary.
2. (Optional) To preserve the continuity of source and target structures, stop DDL activities and then make certain that Replicat has processed all of the DDL and replicated DML data in the trail. To determine when Replicat is finished, issue the following command until you see a message that there is no more data to process.

   ```
   INFO REPLICAT <group>
   ```

   > **NOTE** Instead of using INFO Replicat, you can use the EVENTACTIONS option of TABLE and MAP to stop the Extract and Replicat processes after the DDL and DML has been processed.

3. Run GGSCI.
4. Stop Extract to stop DDL capture.

   ```
   STOP EXTRACT <group>
   ```

5. Stop the Replicat process to stop DDL replication.

   ```
   STOP REPLICAT <group>
   ```

6. Change directories to the GoldenGate installation directory.
7. Run SQL*Plus and log in as a user that has SYSDBA privileges.
8. Disable all sessions that ever issued DDL. Otherwise the database might generate an ORA-04021 error.
9. Run the ddl_disable script to disable the DDL trigger.
10. Run the ddl_remove script to remove the GoldenGate DDL trigger, the DDL history and marker tables, and other associated objects. This script produces a ddl_remove_spool.txt file that logs the script output and a ddl_remove_set.txt file that logs current user environment settings in case they are needed for debugging.
11. Run the marker_remove script to remove the GoldenGate marker support system. This script produces a marker_remove_spool.txt file that logs the script output and a marker_remove_set.txt file in case they are needed for debugging.
12. Run the marker_setup script to reinstall the GoldenGate marker support system. You will be prompted for the name of the GoldenGate schema.
13. Run the ddl_setup script. You will be prompted for:
    - The name of the GoldenGate DDL schema.
    - The installation mode. To reinstall DDL objects, use the INITIALSETUP mode. This mode drops and recreates existing DDL objects before creating new objects.

14. Run the role_setup script to recreate the GoldenGate DDL role.
15. Grant the role you just created to all GoldenGate users under which the following GoldenGate processes run: Extract, Replicat, GGSCI, and Manager. You might need to make multiple grants if the processes have different user names.
16. Run the ddl_enable.sql script to enable the DDL trigger.

Removing the DDL objects from the system

This procedure removes the DDL environment and removes the history that maintains continuity between source and target DDL operations.

NOTE Due to object interdependencies, all objects must be removed.

1. Run GGSCI.
2. Stop Extract to stop DDL capture.

   ```
   STOP EXTRACT <group>
   ```

3. Stop the Replicat process to stop DDL replication.

   ```
   STOP REPLICAT <group>
   ```

4. Change directories to the GoldenGate installation directory.
5. Run SQL*Plus and log in as a user that has SYSDBA privileges.
6. Disable all sessions that ever issued DDL. Otherwise the database might generate an ORA-04021 error.
7. Run the ddl_disable script to disable the DDL trigger.
8. Run the ddl_remove script to remove the GoldenGate DDL trigger, the DDL history and marker tables, and the associated objects. This script produces a ddl_remove_spool.txt file that logs the script output and a ddl_remove_set.txt file that logs current user environment settings in case they are needed for debugging.
9. Run the marker_remove script to remove the GoldenGate marker support system. This script produces a marker_remove_spool.txt file that logs the script output and a marker_remove_set.txt file in case they are needed for debugging.

CHAPTER 6

Uninstalling GoldenGate

This procedure assumes that you no longer need the data in the GoldenGate trails, and that you no longer need to preserve the current GoldenGate environment. To preserve your current environment and data, make a backup of the GoldenGate directory and all subdirectories before starting this procedure.

Uninstalling GoldenGate from UNIX

1. Run the command shell.
2. (Suggested) Log on as the system administrator, or as a user with permission to issue GoldenGate commands, and to delete files and directories from the operating system.
3. Change directories to the GoldenGate installation directory.
4. Run GGSCI.
5. Stop all GoldenGate processes.
6. Stop the Manager process.
7. Exit GGSCI.
8. Remove the GoldenGate files by removing the installation directory.
9. Drop any GoldenGate-related objects from the database as needed.

Uninstalling GoldenGate from Windows (non-cluster)

1. (Suggested) Log on as the system administrator, or as a user with permission to issue GoldenGate commands, and to delete files and directories from the operating system.
2. From the GoldenGate installation folder, run GGSCI.
3. Stop all GoldenGate processes.
4. Stop the Manager program or service.
5. Exit GGSCI.
6. Click **Start** > **Run**, and type cmd in the **Run** dialog box.
7. Change directories to the GoldenGate installation directory.

8. Run the install program using the following syntax.

   ```
   install deleteevents deleteservice
   ```

   This command deletes GoldenGate events from being reported to the Windows Event Manager and removes the GoldenGate Manager service.
9. Delete the CATEGORY.DLL and GGSMSG.DLL files from the Windows SYSTEM32 folder.
10. Delete the GoldenGate installation folder.
11. Drop any GoldenGate-related objects from the database as needed.

Uninstalling GoldenGate from Windows Cluster

1. Working from the node in the cluster that owns the cluster group containing the Manager resource, run GGSCI and then stop any Extract and Replicat processes that are still running.
2. Use the Cluster Administrator tool to take the Manager resource offline.
3. Right click the resource and select **Delete** to remove it.
4. Run the install program using the following syntax.

   ```
   install deleteevents deleteservice
   ```

   This command deletes GoldenGate events from being reported to the Windows Event Manager and removes the GoldenGate Manager service.
5. Delete the CATEGORY.DLL and GGSMSG.DLL files from the Windows SYSTEM32 folder.
6. Move the cluster group to the next node in the cluster, and repeat from step 4.
7. Delete the GoldenGate installation folder.
8. Drop any GoldenGate-related objects from the database as needed.

APPENDIX 1
GoldenGate installed components

This appendix describes the programs, directories, and other components created or used by the GoldenGate software in the GoldenGate installation directory. Additional files not listed here might be installed on certain platforms. Files listed here might not be installed on every platform.

GoldenGate Programs and Utilities

This section describes programs installed in the root GoldenGate installation directory.

Table 10 Programs and utilities

Program	Description
cobgen	Generates source definitions based on COBOL layouts. Used for GoldenGate for Datawise on Stratus.
convchk	Converts checkpoint files to a newer version.
ddlcob	Generates target DDL table creation statements based on COBOL layouts. Used for GoldenGate for Datawise on Stratus.
ddlgen	Generates target database table definitions based on source database DDL.
defgen	Generates data definitions and is referenced by GoldenGate processes when source and target tables have dissimilar definitions.
emsclnt	Sends event messages created by Collector and Replicat on Windows or UNIX systems to EMS on NonStop systems.
extract	Performs extraction from database tables or transaction logs or receives transaction data from a vendor access module.
ggmxinstall	GoldenGate installation script for SQL/MX.
ggsci	User interface to GoldenGate for issuing commands and managing parameter files.

Table 10 Programs and utilities (continued)

Program	Description
ggsmgr.jcl ggsmgr.proc ggsmgrst.jcl ggsmgrst.proc	Start the GoldenGate Manager process from a batch job or the operator console on a z/OS system.
install	Installs GoldenGate as a Windows service and provides other Windows-based service options.
keygen	Generates data-encryption keys.
logdump	A utility for viewing and saving information stored in extract trails or files.
mgr	(Manager) Control process for resource management, control and monitoring of GoldenGate processes, reporting, and routing of requests through the GGSCI interface.
replicat	Applies data to target database tables.
reverse	A utility that reverses the order of transactional operations, so that Replicat can be used to back out changes from target tables, restoring them to a previous state.
server	The Collector process, an Extract TCP/IP server collector that writes data to remote trails.
triggen	Generates scripts that create the GoldenGate log table and logging triggers to support the trigger-based extraction method.
vamserv	Started by Extract to read the TMF audit trails generated by TMF-enabled applications using the NonStop SQL/MX database.

GoldenGate subdirectories

This section describes the subdirectories of the GoldenGate installation directory and their contents.

Table 11 Subdirectories

Directory	Description
dirchk	Contains the checkpoint files created by Extract and Replicat processes, which store current read and write positions to support data accuracy and fault tolerance. Written in internal GoldenGate format. File name format is <group name><sequence number>.<ext> where <sequence number> is a sequential number appended to aged files and <ext> is either cpe for Extract checkpoint files or cpr for Replicat checkpoint files. Do not edit these files. Examples: ext1.cpe rep1.cpr
dirdat	The default location for GoldenGate trail files and extract files created by Extract processes to store records of extracted data for further processing, either by the Replicat process or another application or utility. Written in internal GoldenGate format. File name format is a user-defined two-character prefix followed by either a six-digit sequence number (trail files) or the user-defined name of the associated Extract process group (extract files). Do not edit these files. Examples: rt000001 finance
dirdef	The default location for data definitions files created by the DEFGEN utility to contain source or target data definitions used in a heterogeneous synchronization environment. Written in external ASCII. File name format is a user-defined name specified in the DEFGEN parameter file. These files may be edited to add definitions for newly created tables. If you are unsure of how to edit a definitions file, contact GoldenGate technical support. Example: defs.dat
dirout	This directory is not used any more.

Table 11 Subdirectories (continued)

Directory	Description
dirpcs	Default location for status files. File name format is <group>.<extension> where <group> is the name of the group and <extension> is either pce (Extract), pcr (Replicat), or pcm (Manager). These files are only created while a process is running. The file shows the program name, the process name, the port number, and the process ID. Do not edit these files. Examples: mgr.pcm ext.pce
dirprm	The default location for GoldenGate parameter files created by GoldenGate users to store run-time parameters for GoldenGate process groups or utilities. Written in external ASCII format. File name format is <group name/user-defined name>.prm or mgr.prm. These files may be edited to change GoldenGate parameter values. They can be edited directly from a text editor or by using the EDIT PARAMS command in GGSCI. Examples: defgen.prm finance.prm
dirrec	Not used by GoldenGate.
dirrpt	The default location for process report files created by Extract, Replicat, and Manager processes to report statistical information relating to a processing run. Written in external ASCII format. File name format is <group name><sequence number>.rpt where <sequence number> is a sequential number appended to aged files. Do not edit these files. Examples: fin2.rpt mgr4.rpt
dirsql	The default location for scripts created by the TRIGGEN utility to contain SQL syntax for creating GoldenGate logging triggers and GoldenGate log tables. Written in external ASCII format. File name format is a user-defined name or the defaults of GGSLOG (table-creation script) or the table name (trigger-creation script), with the extension of .sql. These scripts can be edited if needed. Examples: ggslog.sql account.sql

Table 11 Subdirectories (continued)

Directory	Description
dirtmp	The default location for storing large transactions when the size exceeds the allocated memory size. Do not edit these files.
dirver	A GoldenGate Veridata directory. Not used unless this software is installed in the GoldenGate location.

Other GoldenGate files

This section describes other files, templates, and other objects created or installed in the root GoldenGate installation directory.

Table 12 Other files

Component	Description
bcpfmt.tpl	Template for use with Replicat when creating a run file for the Microsoft BCP/DTS bulk-load utility.
blowfish.txt	Blowfish encryption software license agreement.
category.dll	Windows dynamic link library used by the INSTALL program.
chkpt_<db>_create.sql	Script that creates a checkpoint table in the local database. A different script is installed for each database type.
db2cntl.tpl	Template for use with Replicat when creating a control file for the IBM LOADUTIL bulk-load utility.
ddl_access.tpl	Template used by the DDLGEN utility to convert source DDL to Microsoft Access DDL.
ddl_cleartrace.sql	Script that removes the DDL trace file. (Oracle installations)
ddl_db2.tpl	Template used by the DDLGEN utility to convert source DDL to DB2 DDL (Linux, UNIX, Windows).
ddl_db2_os390.tpl	Template used by the DDLGEN utility to convert source DDL to DB2 DDL (z/OS systems).
ddl_disable.sql	Script that disables the GoldenGate DDL trigger. (Oracle installations)
ddl_enable.sql	Script that enables the GoldenGate DDL trigger. (Oracle installations)

Table 12 Other files (continued)

Component	Description
ddl_informix.tpl	Template used by the DDLGEN utility to convert source DDL to Informix DDL.
ddl_mss.tpl	Template used by the DDLGEN utility to convert source DDL to SQL Server DDL.
ddl_mysql.tpl	Template used by the DDLGEN utility to convert source DDL to MySQL DDL.
ddl_nssql.tpl	Template used by the DDLGEN utility to convert source DDL to NonStop SQL DDL.
ddl_ora9.sql	A script that gets tablespace information from an Oracle 9 database.
ddl_ora10.sql	A script that disables the Oracle recyclebin and gets tablespace information from an Oracle 10 database.
ddl_oracle.tpl	Template used by the DDLGEN utility to convert source DDL to Oracle DDL.
ddl_pin.sql	Script that pins DDL tracing, the DDL package, and the DDL trigger for performance improvements. (Oracle installations)
ddl_remove.sql	Script that removes the DDL extraction trigger and package. (Oracle installations)
ddl_setup.sql	Script that installs the GoldenGate DDL extraction and replication objects. (Oracle installations)
ddl_sqlmx.tpl	Template used by the DDLGEN utility to convert Tandem Enscribe DDL to NonStop SQL/MX DDL.
ddl_status.sql	Script that verifies whether or not each object created by the GoldenGate DDL support feature exists and is functioning properly. (Oracle installations)
ddl_sybase.tpl	Template used by the DDLGEN utility to convert source DDL to Sybase DDL.
ddl_tandem.tpl	Template used by the DDLGEN utility to convert source DDL to NonStop SQL DDL.
ddl_tracelevel.sql	Script that sets the level of tracing for the DDL support feature. (Oracle installations)
debug files	Debug text files that may be present if tracing was turned on.

Table 12 Other files (continued)

Component	Description
demo_<db>_create.sql	Script that creates demonstration tables in the database associated with the GoldenGate installation.
demo_<db>_insert.sql	Script that inserts initial test data into the demonstration tables.
demo_<db>_misc.sql	Script that simulates transaction activity on the demonstration tables.
ENCKEYS	User-created file that stores encryption keys. Written in external ASCII format.
exitdemo.c	User exit example.
ggmessage.dat	Data file that contains error, informational, and warning messages that are returned by the GoldenGate processes. The version of this file is checked upon process startup and must be identical to that of the process in order for the process to operate.
ggserr.log	File that logs processing events, messages, errors, and warnings generated by GoldenGate.
ggsmsg.dll	Windows dynamic link library used by the INSTALL program.
GLOBALS	User-created file that stores parameters applying to the GoldenGate instance as a whole.
help.txt	Help file for the GGSCI command interface.
LGPL.txt	Lesser General Public License statement. Applies to free libraries from the Free Software Foundation.
libodbc.so	ODBC file for Ingres 2.6 on Unix.
libodbc.txt	License agreement for libodbc.so.
libxml2.dll	Windows dynamic link library containing the XML library for GoldenGate's XML procedures.
libxml2.txt	License agreement for libxml2.dll.
marker.hist	File created by Replicat if markers were passed from a NonStop source system.
marker_remove.sql	Script that removes the DDL marker table. (Oracle installations)
marker_setup.sql	Script that installs the GoldenGate DDL marker table. (Oracle installations)

Table 12 Other files (continued)

Component	Description
marker_status.sql	Script that confirms successful installation of the DDL marker table. (Oracle installations)
odbcinst.ini	Ingres 2.6 on Unix ODBC configuration file.
params.sql	Script that contains configurable parameters for DDL support. (Oracle installations)
pthread-win32.txt	License agreement for pthread-VC.dll.
pthread-VC.dll	POSIX threads library for Microsoft Windows.
role_setup.sql	Script that creates the database role necessary for GoldenGate DDL support. (Oracle installations)
sampleodbc.ini	Sample ODBC file for Ingres 2.6 on UNIX.
sqlldr.tpl	Template for use with Replicat when creating a control file for the Oracle SQL*Loader bulk-load utility.
start.prm stop.prm	z/OS paramlib members to start and stop the Manager process.
startmgr stopmgr	z/OS Unix System Services scripts to start the Manager process from GGSCI.
startmgrcom stopmgrcom	z/OS system input command for the Manager process.
tcperrs	File containing user-defined instructions for responding to TCP/IP errors.
usrdecs.h	Include file for user exit API.
zlib.txt	License agreement for zlib compression library.

GoldenGate checkpoint table

When database checkpoints are being used, GoldenGate creates a checkpoint table with a user-defined name in the database upon execution of the ADD CHECKPOINTTABLE command, or a user can create the table by using the chkpt_<db>_create.sql script, where <db> is the type of database.

Do not change the names or attributes of the columns in this table. You can change table storage attributes as needed.

Table 13 Checkpoint table definitions

Column	Description
GROUP_NAME (primary key)	The name of a Replicat group using this table for checkpoints. There can be multiple Replicat groups using the same table.
GROUP_KEY (primary key)	A unique identifier that, together with GROUPNAME, uniquely identifies a checkpoint regardless of how many Replicat groups are writing to the same table.
SEQNO	The sequence number of the checkpoint file.
RBA	The relative byte address of the checkpoint in the file.
AUDIT_TS	The timestamp of the checkpoint position in the checkpoint file.
CREATE_TS	The date and time when the checkpoint table was created.
LAST_UPDATE_TS	The date and time when the checkpoint table was last updated.
CURRENT_DIR	The current GoldenGate home directory or folder.

Index

Symbols

$LD_LIBRARY_PATH variable 26
$PATH variable 26

A

ADD TRANDATA command 49
ADDEVENTS Windows service option 29
ADDSERVICE Windows service option 30
ALTARCHIVEDLOGFORMAT option, TRANLOGOPTIONS 43
ALTARCHIVELOGDEST option, TRANLOGOPTIONS 43
ANYDATA datat type 15
ANYDATASET data type 15
ANYTYPE data type 15
APPEND hint 19
archive logs, configuring 41
ASMPASSWORD option, TRANLOGOPTIONS 9
ASMUSER option, TRANLOGOPTIONS 9
Automatic Storage Management (ASM) 9, 40
AUTOSTART Windows service option 30

B

BASICFILE LOB 12
BFILE data type 15
binary data types 12
BINARY DOUBLE data type 11
BINARY FLOAT data type 11
BINARY_INTEGER 15
BINARY_INTEGER data type 15
BINDCPU option, THREADOPTIONS 55
BLOB data type 12
BUFFER hint 19

C

cascade deletes, disabling 48
case, supported 20
category.dll 29
chained rows 44
CHAR data type 11
character data types 11
character set, configuring 40
characters
  multibyte 11
  supported in object names 21
CLOB data type 12
clocks, synchronizing 7, 55
cluster, installing on 6, 27, 28, 30
clustered database, installing on 7
clustered tables 17
code points of character set, representing 12
collection types, support for 14
COLS option, ADD TRANDATA 50
columns
  number and size supported 15
  unused 16
  virtual 16
components, GoldenGate 66
COMPRESS option of CREATE 19
compression, table 16
connections, to database 9
constraint checking, deferring 51
constraints, integrity 48
CONVERTUCS2CLOBS option, TRANLOGOPTIONS 12
CREATE SUBDIRS command 27
cursors, open 44

D

database
multiple instances 6
preparing for GoldenGate 40
requirements 8
versions supported 5
Database Replay 19
date and time data types 12
DATE data type 12
DBLOGIN command 50
DBMS_LOB function 12
DDL
installing support for 36
managing replication environment 57
nonsupported objects and operations 19
supported objects and operations 18
ddl_cleartrace script 59
ddl_disable script 57
ddl_enable script 57
ddl_pin script 39
ddl_remove script 62, 63
ddl_remove_ files 63
ddl_remove_ spool 62
ddl_setup script 38, 60, 61
deletes, cascaded 48
deleting
DDL history table 58
DDL marker table 58
DDL objects from system 63
DDL trace file 59
DICOM, support for 15
direct load 19
disk requirements 6
downloading GoldenGate 23

E

environment variables, setting 24, 25
escape sequence for Unicode data 12
EXTERNAL attribute of tables 16

F

fetch options, setting 46
FETCHCHAINEDUPDATES option, TRANLOGOPTIONS 50
FETCHOPTIONS parameter 48
files installed by GoldenGate 66
firewall, configuring 7
fixed-point numbers 11
Flashback Query, using 47
floating-point numbers 11
FORCETRANS option, SEND EXTRACT 45
FTP access for GoldenGate 7

G

GEORASTER, capturing 52
ggmessage.dat file 72
GGS_ tables for DDL support 36
ggs_ddl_trace log 59
GGSMGR default Manager name 28
ggsmsg.dll 29
globalization settings, viewing and configuring 40
GLOBALS file 29
GoldenGate
downloading 23
installed programs and files 66
installing 23
uninstalling 64

H

HANDLETPKUPDATE parameter 51

I

I/O on redo logs 41
INQUEUESIZE option, THREADOPTIONS 56

installing on
Linux and UNIX 27
Windows 28
INTERVAL data type 15
interval partitioning 16
Itanium preinstallation requirements 8

K

key
absence of 50
assigning 48
including in redo record 49
name, supported characters 21
transient updates 51
KEYCOLS option, TABLE or MAP 49

L

large objects, limitations on 13
LIBPATH variable 26
libraries, Visual C++ 8, 28
LIMITROWS option, DBOPTIONS 50
Linux, installing on 27
LOB data types
retention, setting 47
support for 12
log retention, calculating 42
LOG_ARCHIVE_FORMAT parameter 43
LOG_PARALLELISM parameter 41
logs, configuring 41
LOGSOURCE option, TRANLOGOPTIONS 42
LONG data types 11, 12

M

Manager
as Windows service 29
multiple on same system 28
name, customizing 28
MANUALSTART Windows service option 30
marker_remove script 62, 63
marker_remove_spool file 62, 63
marker_setup script 38
materialized views 17
MAXFETCHSTATEMENTS parameter 44, 48
Memory requirements for GoldenGate 5
MGRSERVNAME parameter 29
Microsoft Visual C ++ 2005 SP1 Redistributable Package 8, 28
MLSLABEL data type 15
multi-byte data types 11
multimedia types, support for 15
multiple database instances 6

N

name
non-supported characters in 22
supported characters in 21
names, supported 20
NCHAR data type 11
NCLOB data type 12
nested table 14
NLS_LANG parameter 41
NLS_NCHAR_CHARACTERSET parameter 12
NOKEY option, ADD TRANDATA 50
NUMBER data type 11
numeric data types 11
NVARCHAR2 data type 11

O

object tables, support for 14
objects, supported 15
OCILobWrite function 12
operating systems supported 5
operations, supported 15
ORACLE_HOME and ORACLE_SID settings 24
Oracle, versions supported 5
ORDDicom type, support for 15
OUTQUEUESIZE option, THREADOPTIONS 56

P

parallelism, redo logs 41
partitioning, interval 16
PASSWORD Windows service option 30
PATHMAP option, TRANLOGOPTIONS 42

PCTVERSION parameter 47
platforms, supported 5
PLS_INTEGER data type 15
ports, required by GoldenGate 7
pre-installation instructions 5
privileges
database 9
operating system 8
PURGEDDLHISTORY parameter 58
PURGEMARKERHISTORY parameter 57
PURGEORPHANEDTRANSACTIONS option, TRANLOGOPTIONS 56

Q

queries
on undo tablespace 47
prepared, number of 48

R

RAID drives 42
RAW data type 12
Real Application Cluster 7
recyclebin, Oracle 19
redo logs, configuring 41
REFs 19
removing
DDL objects 63
GoldenGate from system 64
REPFETCHEDCOLOPTIONS parameter 48
REPORTFETCH option, STATOPTIONS 47, 48
role_setup script 39
ROWID data type 15
rows
chained 44
number and size supported 15

S

SDO_GEOMETRY type 14
SDO_GEORASTER type 14
SDO_TOPO_GEOMETRY type 14
SECUREFILE LOB 12, 13
semantics, considerations for 12
sequences, supported 18
SETENV parameter 24
SHLIB_PATH variable 26
SHOWTRANS option, SEND EXTRACT 45
SKIPTRANS option, SEND EXTRACT 45
snapshot, using for fetches 47
spaces
in folder names 28
in object and column names 22
Spatial types
capturing 52
support for 14
SQL statements, prepared 44
subdirectories, creating 27
synonyms 19
system requirements 5

T

tables
clustered 17
created as EXTERNAL 16
preparing for processing 48
supported types 15
TCP/IP, configuring 7
THREADOPTIONS with BINDCPU 55
TIMESTAMP data type 12
TIMEZONE_ABBR 15
TIMEZONE_REGION 15
transaction logs, configuring 41
transactions, long-running 45
TRANSCLEANUPFREQUENCY option, TRANLOGOPTIONS 56
transient primary key update 51
Transparent Data Encryption 51
triggers, disabling on target 48

U

undo tablespace, Oracle 47
UNDO_MANAGEMENT parameter 47
UNDO_RETENTION parameter 47
uninstalling GoldenGate 64

UNIX, installing on 27
URITYPE data type 15
UROWID data type 15
user defined type 13
USER Windows service option 30
user, GoldenGate 9
USERID parameter 9
USETRIGGER option, ADD TRANDATA 50

V

VAMSERV program 67
VARCHAR2 data type 11
VARRAY data type 15
VARWIDTHNCHAR parameter 12
vcredist_IA64.exe runtime library 8
versions of Oracle supported 5
views 17
virtual columns 16
virtual machine, support for 8
Visual C ++ 2005 SP1 Redistributable Package 8, 28

W

WARNLONGTRANS parameter 45
Windows, installing on 28

X

XMLType 13