Oracle® GoldenGate

Troubleshooting and Tuning Guide

Version 10.4

October 2009

Troubleshooting and Tuning Guide, version 10.4

Contents

PREFACE

About the GoldenGate Guides

The complete GoldenGate documentation set contains the following components:

Oracle® GoldenGate for Mainframe

- *GoldenGate for HP NonStop Administrator Guide:* Introduces GoldenGate components and explains how to plan for, configure, and implement GoldenGate on the NonStop platform.
- *GoldenGate for HP NonStop Reference Guide:* Provides detailed information about GoldenGate parameters, commands, and functions for the NonStop platform.

Windows and UNIX platforms

- *Installation and Setup guides*: There is one such guide for each database that is supported by GoldenGate.
- *GoldenGate for Windows and UNIX Administrator Guide*: Introduces GoldenGate components and explains how to plan for, configure, and implement GoldenGate on the Windows and UNIX platforms.
- *GoldenGate for Windows and UNIX Reference Guide*: Provides detailed information about GoldenGate parameters, commands, and functions for the Windows and UNIX platforms.
- *Troubleshooting and Tuning Guide:* Provides suggestions for improving the performance of GoldenGate in different situations, and provides solutions to common problems.

Typographic conventions used in this manual

This manual uses the following style conventions.

- Parameter and command arguments are shown in upper case, for example:

  ```
  CHECKPARAMS
  ```

- File names, table names, and other names are shown in lower case unless they are case-sensitive to the operating system or software application they are associated with, for example:

  ```
  account_tab
  GLOBALS
  ```

- Variables are shown within < > characters, for example:

  ```
  <group name>
  ```

- When one of multiple mutually-exclusive arguments must be selected, the selection is enclosed within braces and separated with pipe characters, for example:

  ```
  VIEW PARAMS {MGR | <group> | <file name>}
  ```

- Optional arguments are enclosed within brackets, for example:

  ```
  CLEANUP EXTRACT <group name> [, SAVE <count>]
  ```

- When there are numerous multiple optional arguments, a placeholder such as [<option>] may be used, and the options are listed and described separately, for example:

  ```
  TRANLOGOPTIONS [<option>]
  ```

- When an argument is accepted more than once, an ellipsis character (...) is used, for example:

  ```
  PARAMS ([<requirement rule>] <param spec> [, <param spec>] [, ...])
  ```

New in this version of GoldenGate

Database-specific enhancements

DB2 LUW

- GoldenGate now supports Multi Dimensional Clustered Tables (MDC) for DB2 LUW 9.5 and later.
- GoldenGate now supports Materialized Query Tables. GoldenGate does not replicate the MQT itself, but only the base tables. The target database automatically maintains the content of the MQT based on the changes that are applied to the base tables by Replicat.

DB2 z/OS

A new SQLID option was added to DBLOGIN that issues the SQL command SET CURRENT SQLID = 'sqlid' after the USERID login (with PASSWORD, if applicable) is completed. If the SET command fails, the entire DBLOGIN command fails as a unit.

Oracle

- GoldenGate now supports Oracle 10.2 running on zLinux SUSE on S/390. A new LOGSOURCE target of S390 has been added to TRANLOGOPTIONS.
- For Oracle versions 10g and later, you can now use a simple parameter statement instead of a trace table to prevent Replicat transactions from being captured in a bi-directional configuration. This enhancement eliminates the overhead of creating and writing to the trace table. The new EXCLUDEUSER and EXCLUDEUSERID options of TRANLOGOPTIONS support this enhancement. These options also can be used to exclude the work of other database users, and multiple instances of either parameter can be used. For more information, see the *GoldenGate for Windows and UNIX Reference Guide*. For Oracle versions 9i and earlier, the trace table is still required.

- Concurrent with the EXCLUDEUSER and EXCLUDEUSERID enhancements, @GETENV was enhanced with two new functions: @GETENV ("TRANSACTION", "USERID") returns the user-id for the Oracle user that issued the last committed transaction. @GETENV ("TRANSACTION", "USERNAME") returns the name of the Oracle user that issued the last committed transaction.
- GoldenGate now supports Oracle Spatial objects, object tables, Oracle Multimedia ORDDicom (DICOM), and XMLType stored as an object. A new XMLBUFSIZE option to DBOPTIONS sets the size of the memory buffer that stores XML data that was extracted from the sys.xmltype attribute of a SDO_GEORASTER object type.
- GoldenGate now supports cluster tables.
- GoldenGate now supports the capture and replication of Oracle DDL statements of up to 2 MB in length (including some internal GoldenGate maintenance information). Extract will skip statements that are greater than the supported length, but a new ddl_ddl2file.sql script can be used to save the skipped DDL to a text file in the USER_DUMP_DEST directory of Oracle.

  NOTE To use the new support, the DDL trigger must be reinstalled in INITIALSETUP mode, which removes all of the DDL history. Follow the procedure for restoring an existing DLL environment to a clean state in the Oracle Installation and Setup Guide.

- A new INSERTAPPEND and NOINSERTAPPEND parameter set controls whether or not Replicat uses an APPEND hint when it applies inserts to Oracle target tables. INSERTAPPEND is appropriate for use as a performance improvement when the replicated transactions are large and contain multiple inserts into the same table. These parameters can be used in two ways: When used as standalone parameters at the root of the parameter file, one remains in effect for all subsequent TABLE or MAP statements, until the other is encountered. When used within a MAP statement, they override any standalone INSERTAPPEND or NOINSERTAPPEND entry that precedes the MAP statement.
- Schema name translation for DDL operations on objects of UNMAPPED scope was revised. When Oracle DDL is of UNMAPPED scope in the Replicat configuration, Replicat will set the current owner of the Replicat session to the owner of the source DDL object, then execute the DDL as that owner, then restore Replicat as the current owner of the Replicat session.

SQL Server

GoldenGate now supports delivery to a SQL Server 2008 target. See the GoldenGate for SQL Server Installation and Setup Guide for more information.

Teradata

Database name translation for DDL operations on objects of UNMAPPED scope was revised. When Teradata DDL is of UNMAPPED scope in the Replicat configuration, it is applied to the target in one of these ways:

- If the required Replicat connection parameter TARGETDB contains just a DSN (as in tdtarg), but not a database name, it is applied to the target object with the same owner (database name) and object name as in the source DDL.
- If a specific database name is used in TARGETDB (as in db@tdtarg), all of the DDL operations are applied to the target with the owner from TARGETDB.

Other enhancements

General

- GoldenGate now supports the replication of DDL statements that contain a space before or after the owner and object names (or both before and after). For example, fin . customers.

Column Conversion Functions

- A new OSVARIABLE option of @GETENV returns the string value of a specified operating-system environment variable. For example, @GETENV ("OSVARIABLE", "HOME") would return the value of the HOME variable. Because the function only returns an exact match, other variables that contain "HOME," such as ORACLE_HOME, would not be returned. This option is valid for Extract and Replicat.

Parameters

- A new TRANSACTIONTIMEOUT parameter for Replicat limits the amount of time that Replicat will hold a target transaction open if it has not received the end-of-transaction record for the last source transaction in that transaction. TRANSACTIONTIMEOUT helps prevent an uncommitted Replicat target transaction from holding locks on the target database and consuming its resources unnecessarily. You can change the value of this parameter so that Replicat can work within existing application timeouts and other database requirements on the target.
- A new GLOBALS parameter OUTPUTFILEUMASK specifies an octal umask that will be used by GoldenGate processes to create trail files and discard files. This parameter is not valid for WIN32 systems.
- The default for CACHESIZE is now 8GB for 64-bit systems and 2GB for 32-bit systems.

Commands

- The SEND EXTRACT <group> STATUS command now makes it clear when Extract is going through a recovery after an abend event. This feature is especially useful when a very long-running transaction was open at the point of failure and Extract must search far back into the logs to find the begin-transaction record. The new status messages appear on the Current status line and look as follows:
  - In recovery[1] – Extract is recovering to its input checkpoint.
  - In recovery[2] – Extract is recovering to its output checkpoint.
  - Recovery complete – The recovery is finished, and normal processing will resume.

- To support TRANSACTIONTIMEOUT, the SEND REPLICAT command with STATUS has two more status conditions:
  - Performing transaction timeout recovery – Aborting current incomplete transaction and repositioning to start new one (see TRANSACTIONTIMEOUT parameter).
  - Waiting for data at logical EOF after transaction timeout recovery – Waiting to receive remainder of incomplete source transaction after a TRANSACTIONTIMEOUT termination.

Documentation enhancements and corrections

The documentation for the @GETENV options DBTRANSACTION and DBRECORD was changed to reflect the accurate names of TRANSACTION and RECORD. Various other minor errors were corrected in this section.

The FORMAT option was added to the RMTTASK documentation. The same versioning concept that applies to trails and files also applies to the data format that is sent by Extract to a remote process.

The default for AUTORESTART RESETMINUTES was changed to the correct value of 20 minutes.

CHAPTER 1

GoldenGate performance and troubleshooting tools

Overview of the GoldenGate performance tools

GoldenGate provides a variety of tools and resources that help you:

- Monitor processing activity
- Tune performance
- Solve problems

This information is obtained through:

- GGSCI commands
- GoldenGate tracing parameters
- GoldenGate reports and logs
- GoldenGate Director graphical user interface
- System logs
- GoldenGate Logdump utility
- GoldenGate error documentation
- GoldenGate Knowledge Base

Using the information commands in GGSCI

The primary way to view processing information is through GGSCI. For more information about these commands, see the *GoldenGate for Windows and UNIX Reference Guide*.

Table 1 Commands to view process information

Command	What it shows
INFO {EXTRACT | REPLICAT} <group> [DETAIL]	Run status, checkpoints, approximate lag, and environmental information
INFO MANAGER	Run status and port number
INFO ALL	INFO output for all GoldenGate processes on the system

Table 1 Commands to view process information

Command	What it shows
STATS {EXTRACT | REPLICAT} <group>	Statistics for operations processed
STATUS {EXTRACT | REPLICAT} <group>	Run status (starting, running, stopped, abended)
STATUS MANAGER	Run status
LAG {EXTRACT | REPLICAT} <group>	Latency between last record processed and timestamp in the data source
INFO {EXTTRAIL | RMTTRAIL} <path name>	Name of associated process, position of last data processed, maximum file size
SEND MANAGER	Run status, information about child processes, port information, trail purge settings
SEND {EXTRACT | REPLICAT}	Depending on the process, returns information about memory pool, lag, TCP statistics, long-running transactions, process status, recovery progress, and more.
VIEW REPORT <group>	Contents of the process report
VIEW GGSEVT	Contents of the GoldenGate error log

<command> ER <wildcard>	Information dependent on the <command> type:
	INFO
	LAG
	SEND
	STATS
	STATUS
	<wildcard> is a wildcard specification for the process groups to be affected, for example:
	`INFO ER ext*`
	`STATS ER *`

Positioning processes using GGSCI commands

In the course of resolving a problem, you might need to position an Extract process in the transaction log or in an extract trail, or to position Replicat in the trail, such as to skip data. The following commands have reposition options:

- ADD EXTRACT and ALTER EXTRACT for a primary Extract or a data pump have options to start at a given timestamp or physical location in the data source. In addition, ALTER EXTRACT enables you to start a new file within a trail. (SEND EXTRACT also provides this capability.)

- SEND EXTRACT provides the option to skip the oldest transaction, allowing the management of long-running transactions.
- ADD REPLICAT and ALTER REPLICAT have options to start at a given timestamp or at a physical location in the trail.
- START REPLICAT has options to skip the first transaction from its restart position in the trail, or to start processing at, or after, a transaction indicator that contains a given commit sequence number (CSN).

For full details and syntax, see the *GoldenGate for Windows and UNIX Reference Guide*.

Using the process report

Use the process report to view (depending on the process):

- parameters in use
- table and column mapping
- database information
- runtime messages and errors
- runtime statistics for the number of operations processed

Every Extract, Replicat, and Manager process generates a report file at the end of each run. The report can help you diagnose problems that occurred during the run, such as invalid mapping syntax, SQL errors, and connection errors.

Figure 1 Sample Extract process report

```
********************************************************************
** Running with the following parameters **
********************************************************************
sourceisfile
userid ggs, password ********
rmthost sys1, mgrport 8040
rmtfile /home/ggsora/dirdat/tcustord.dat, purge
table tcustord;

Processing table TCUSTORD

********************************************************************
** Run Time Statistics **
********************************************************************
Report at 2006-08-13 11:07:36 (activity since 2006-08-13 11:07:31)

Output to /home/ggsora/dirdat/tcustord.dat:

From Table TCUSTORD:
       #       inserts: 2
       #       updates:0
       #       deletes:0
       #      discards:0
```

To view a process report

Use any of the following:

- standard shell command for viewing a text file
- GoldenGate Director
- VIEW REPORT command in GGSCI

Syntax

```
VIEW REPORT {<group> | <file name> | MGR}
```

Where:

- <group> shows an Extract or Replicat report that has the default name, which is the name of the associated group.
- <file name> shows any Extract or Replicat report file that matches a given full path name. Must be used if a non-default report name was assigned with the REPORT option of the ADD EXTRACT or ADD REPLICAT command when the group was created.
- MGR shows the Manager process report.

Report names are in upper case if the operating system is case-sensitive. By default, reports have a file extension of .rpt, for example EXTORA.rpt. The default location is the dirrpt sub-directory of the GoldenGate directory.

To determine the name and location of a process report

Use the INFO command in GGSCI.

Syntax

```
INFO <group>, DETAIL
```

To view information if a process abends without a report

Run the process from the command shell of the operating system (not GGSCI) to send the information to the terminal.

Syntax

```
<process> paramfile <path name>.prm
```

Where:

- <process> is either Extract or Replicat.
- paramfile <path name>.prm is the fully qualified name of the parameter file.

Example

```
replicat paramfile /ggs/dirdat/repora.prm
```

Scheduling runtime statistics in the process report

By default, runtime statistics are written to the report once, at the end of each run. For long or continuous runs, you can use optional parameters to view these statistics on a regular basis, without waiting for the end of the run.

To set a schedule for reporting runtime statistics

Use the REPORT parameter in the Extract or Replicat parameter file to specify a day and time to generate runtime statistics in the report.

To send runtime statistics to the report on demand

Use the SEND EXTRACT or SEND REPLICAT command with the REPORT option to view current runtime statistics when needed.

Viewing record counts in the process report

Use the REPORTCOUNT parameter to report a count of transaction records that Extract or Replicat processed since startup. Each transaction record represents a logical database operation that was performed within a transaction that was captured by GoldenGate. The record count is printed to the report file and to the screen.

Managing process reports

Once created, a report file must remain in its original location for GoldenGate to operate properly after processing has started.

Whenever a process starts, GoldenGate creates a new report file and ages the previous one by appending a sequence number to the name. The numbers increment from 0 (the previous one) to 9 (the oldest).

No process ever has more than ten aged reports and one active report. After the tenth aged report, the oldest is deleted when a new report is created. Set up an archiving schedule for aged report files in case they are needed by GoldenGate to resolve a support case.

Figure 2 Current Extract and Manager reports plus aged reports

```
-rw-rw-rw-    1 ggs ggs    1193 Oct 11 14:59    MGR.rpt
-rw-rw-rw-    1 ggs ggs    3996 Oct 5  14:02    MGR0.rpt
-rw-rw-rw-    1 ggs ggs    4384 Oct 5  14:02    TCUST.rpt
-rw-rw-rw-    1 ggs ggs    1011 Sep 27 14:10    TCUST0.rpt
-rw-rw-rw-    1 ggs ggs    3184 Sep 27 14:10    TCUST1.rpt
-rw-rw-rw-    1 ggs ggs    2655 Sep 27 14:06    TCUST2.rpt
-rw-rw-rw-    1 ggs ggs    2655 Sep 27 14:04    TCUST3.rpt
-rw-rw-rw-    1 ggs ggs    2744 Sep 27 13:56    TCUST4.rpt
-rw-rw-rw-    1 ggs ggs    3571 Aug 29 14:27    TCUST5.rpt
```

To prevent an Extract or Replicat report file from becoming too large

Use the REPORTROLLOVER parameter to force report files to age on a regular schedule, instead of when a process starts. For long or continuous runs, setting an aging schedule controls the size of the active report file and provides a more predictable set of archives that can be included in your archiving routine.

To prevent SQL errors from filling up the Replicat report

Use the WARNRATE parameter to set a threshold for the number of SQL errors that can be tolerated on any target table before being reported to the process report and to the error log. The errors are reported as a warning. If your environment can tolerate a large number of these errors, increasing WARNRATE helps to minimize the size of those files.

Using the error log

Use the GoldenGate error log to view:

- a history of GGSCI commands
- GoldenGate processes that started and stopped
- processing that was performed
- errors that occurred
- informational and warning messages

Because the error log shows events as they occurred in sequence, it is a good tool for detecting the cause (or causes) of an error. For example, you might discover that:

- someone stopped a process
- a process failed to make a TCP/IP or database connection
- a process could not open a file

Figure 3 The GoldenGate Error Log (ggserr.log file)

```
2006-01-08 11:20:56  GGS INFO     301  GoldenGate Manager for Oracle,
mgr.prm:  Command received from GUI (START GGSCI ).
2006-01-08 11:20:56  GGS INFO     302  GoldenGate Manager for Oracle,
mgr.prm:  Manager started GGSCI process on port 7840.
2006-01-08 11:21:31  GGS INFO     301  GoldenGate Manager for Oracle,
mgr.prm:  Command received from GUI (START GGSCI ).
2006-01-08 11:21:31  GGS INFO     302  GoldenGate Manager for Oracle,
mgr.prm:  Manager started GGSCI process on port 7841.
2006-01-08 11:24:15  GGS INFO     301  GoldenGate Manager for Oracle,
mgr.prm:  Command received from GUI (START GGSCI ).
2006-01-08 11:24:15  GGS INFO     302  GoldenGate Manager for Oracle,
mgr.prm:  Manager started GGSCI process on port 7842.
2006-01-08 11:24:16  GGS INFO    399  GoldenGate Command Interpreter for
Oracle:  GGSCI command (ggs): add extract extcust  tranlog, begin now.
2006-01-08 11:30:19  GGS INFO    399  GoldenGate Command Interpreter for
Oracle:  GGSCI command (ggs): add rmttrail /home/ggs, extract ggs
```

To view the error log

Use any of the following:

- Standard shell command to view the ggserr.log file within the root GoldenGate directory
- GoldenGate Director
- VIEW GGSEVT command in GGSCI

Syntax `VIEW GGSEVT`

To filter the error log

The error log can become very large, but you can filter it based on a keyword. For example, this filter show only errors:

```
$ more ggserr.log | grep ERROR
```

Because the error log will continue to grow as you use GoldenGate, consider archiving and deleting the oldest entries in the file.

NOTE The Collector process might stop reporting to the log on UNIX systems after the log has been cleaned up. To get reporting started again, restart the Collector process after the cleanup.

Using the discard file

Use a discard file to capture information about GoldenGate operations that failed. This information can help you to resolve data errors, such as those that involve invalid column mapping.

The discard file reports such information as:

- The database error message
- The sequence number of the data source or trail file
- The relative byte address of the record in the data source or trail file
- The details of the discarded operation, such as column values of a DML statement or the text of a DDL statement.

A discard file can be used for Extract or Replicat, but it is most useful for Replicat to log operations that could not be reconstructed or applied.

Figure 4 Sample discard file

```
ORA-20017:  asta0009 6144935
ORA-06512: at "LON.STARTASTA0009_INSERT", line 31
ORA-04088: error during execution of trigger 'LON.STARTASTA0009_INSERT'

Operation failed at seqno 45 rba 12483311
Problem replicating PRODTAB.ASTA0009 to ASTA0009

Error occurred with insert record (target format)...
*
A_TIMESTAMP = 2006-05-15 13:18:32
RELA_PERSON_NR = 3618047
RELA_BEZART = 1
RELA_BEZCODE = 01
RELA_AZ_BAFL = 2819220
RELA_STEMPEL = 0
AKTION = I
OK = 1.0000
NOTOK = -1.0000
*
```

To use a discard file

Include the DISCARDFILE parameter in the Extract or Replicat parameter file. You must supply a name for the file. The parameter has options that control the maximum file size, after which the process abends, and whether new content overwrites or appends to existing content.

Syntax
```
DISCARDFILE <file name> [, APPEND | PURGE] [, MAXBYTES <n> | MEGABYTES <n>]
```

NOTE A process does not create a discard file until the first time that it cannot process a record from its data source.

To view a discard file

Use either of the following:

- Standard shell command to view the file by name
- VIEW REPORT command in GGSCI, with the discard file name as input

Syntax
```
VIEW REPORT <file name>
```

To manage discard files

Use the DISCARDROLLOVER parameter to set a schedule for aging discard files. For long or continuous runs, setting an aging schedule prevents the discard file from filling up and causing the process to abend, and it provides a predictable set of archives that can be included in your archiving routine.

Syntax
```
DISCARDROLLOVER {AT <hh:mi> | ON <day of week> | AT <hh:mi> ON <day of week>}
```

Using GoldenGate Director

Most of the information that can be viewed with GGSCI commands can also be viewed through GoldenGate Director Client and GoldenGate Director Web, GoldenGate's graphical user interfaces. For more information, see the GoldenGate Director online help.

Using the system logs

GoldenGate writes errors that are generated at the level of the operating system to the Event Viewer on Windows or to the syslog on UNIX.

- GoldenGate messaging to the syslog is enabled by default on UNIX.
- On Windows, GoldenGate messaging to the Event Viewer is disabled by default.

To prevent GoldenGate messaging on UNIX

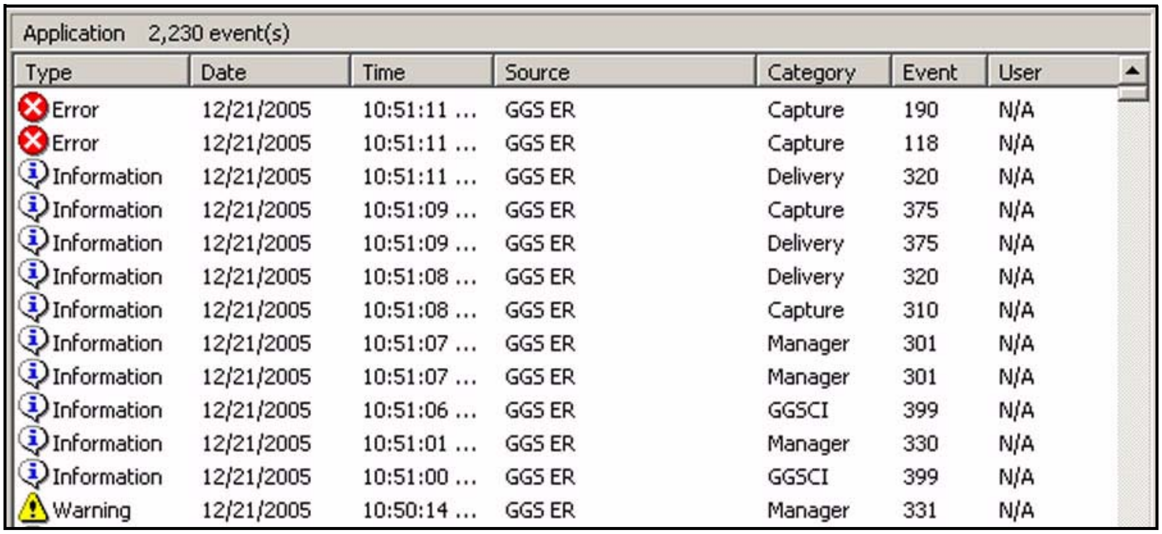
To prevent GoldenGate from logging messages to the syslog, exclude the "user.info" or "*.info" from the syslog file. Contact your Systems Administrator for assistance with this procedure.

To enable GoldenGate messaging on Windows

1. Run the install program with the addevents option. This enables generic messages to be logged.
2. (Optional) To get more specific Windows messages, copy the category.dll and ggsmsg.dll libraries from the GoldenGate directory to the SYSTEM32 directory, either before or after running install. The detailed messages contain the GoldenGate user name and process, the name of the parameter file, and the error text.

NOTE Windows event messaging might have been installed when GoldenGate was installed. For more information on running install, see the GoldenGate installation guide for your database.

Figure 5 GoldenGate messages as seen in the Windows Even Viewer



Application 2,230 event(s)

Type	Date	Time	Source	Category	Event	User
Error	12/21/2005	10:51:11 ...	GGS ER	Capture	190	N/A
Error	12/21/2005	10:51:11 ...	GGS ER	Capture	118	N/A
Information	12/21/2005	10:51:11 ...	GGS ER	Delivery	320	N/A
Information	12/21/2005	10:51:09 ...	GGS ER	Capture	375	N/A
Information	12/21/2005	10:51:09 ...	GGS ER	Delivery	375	N/A
Information	12/21/2005	10:51:08 ...	GGS ER	Delivery	320	N/A
Information	12/21/2005	10:51:08 ...	GGS ER	Capture	310	N/A
Information	12/21/2005	10:51:07 ...	GGS ER	Manager	301	N/A
Information	12/21/2005	10:51:07 ...	GGS ER	Manager	301	N/A
Information	12/21/2005	10:51:06 ...	GGS ER	GGSCI	399	N/A
Information	12/21/2005	10:51:01 ...	GGS ER	Manager	330	N/A
Information	12/21/2005	10:51:00 ...	GGS ER	GGSCI	399	N/A
Warning	12/21/2005	10:50:14 ...	GGS ER	Manager	331	N/A

GoldenGate events in the UNIX system log are similar to those in the Windows log. See your system documentation for more information about the system logs. GoldenGate errors that appear in the system logs also appear in the GoldenGate error log.

Using SHOWSYNTAX to view Replicat SQL statements

Use the SHOWSYNTAX parameter to start an interactive session where you can view each Replicat SQL statement before it is applied. By viewing the syntax of SQL statements that failed, you might be able to diagnose the cause of the problem. For example, you could find out that the WHERE clause is using a non-indexed column.

Using tracing parameters

Use the tracing parameters only as directed by a GoldenGate support analyst when trying to resolve a support case. Tracing captures processing activity that otherwise is not included in the GoldenGate output files. It adds a large amount of overhead to a system, and the output requires expert interpretation.

NOTE Tracing for Teradata installations *must* involve a GoldenGate support analyst, no matter what the circumstances, because it requires a special trace file and debug version of Extract and Replicat.

To trace transaction log activity

Use the TLTRACE parameter to trace the activity of the database transaction log. Two levels of tracing can show either basic or detailed information about DML and DDL operations being processed.

To trace Extract and Replicat processing

Use the TRACE and TRACE2 parameters to capture Extract or Replicat processing information to help reveal processing bottlenecks.

- TRACE provides step-by-step processing information.
- TRACE2 identifies the code segments on which Extract or Replicat is spending the most time.

Both support the tracing of DML and DDL.

Using DUMPDDL to view DDL statements

Use the DUMPDDL command to view the data in the GoldenGate DDL history table. This information is the same information that is used by the Extract process. It is stored in proprietary format, but can be exported in human-readable form to the screen or to a series of SQL tables that can be queried by using regular SQL.

Using GoldenGate error documentation

Use the GoldenGate error documentation in this manual to view descriptions, causes, and solutions for some of the more common error messages that can appear in the GoldenGate process report and in the error log.

Using the GoldenGate Knowledge Base

Before you decide to open a support case, see if you can find a solution to your problem in the GoldenGate Knowledge Base on the GoldenGate support website. This database is updated frequently with new solutions, best practices, workarounds, and other information that is reported by customers, partners, and GoldenGate employees. The interface enables you to perform a simple or complex search, or to send a question to the GoldenGate Technical Services team.

To view the GoldenGate Knowledge Base

Go to http://support.goldengate.com.

To get an ID and password to access the support site

Send an e-mail to ggshelp@goldengate.com.

Using the Logdump utility

Use the Logdump utility to search for, filter, view, and save data that is stored in a GoldenGate trail or extract file. Using Logdump is not necessary to solve most problems, but it is useful when you need to view data in a GoldenGate record. For more information about using Logdump, see Chapter 4.

CHAPTER 2

Tuning GoldenGate's Performance

Overview of performance tuning

One of the challenges of performance tuning is in deciding which component of your GoldenGate environment needs tuning:

- the system or network?
- the database or applications?
- the GoldenGate configuration?

To address these questions, you can tune the performance of the following components in a GoldenGate environment:

- Manager
- Primary Extract process
- Data pump Extract process
- Network
- Replicat

NOTE Performance tuning and troubleshooting often are done at the same time. See "Troubleshooting GoldenGate" on page 35 if you need additional help.

Timing process startup

You can tune the Manager process to start local Extract and Replicat processes automatically and to control the timing of the startup,

To start processes automatically when Manager starts

Use the AUTOSTART parameter to start one or more Extract and Replicat processes automatically when Manager starts. AUTOSTART ensures that no process groups are overlooked and that synchronization activities start immediately.

To restart processes automatically after recovery

Use the AUTORESTART parameter to start one or more Extract and Replicat processes automatically after they fail. AUTORESTART provides fault tolerance when something temporary interferes with a process, such as intermittent network outages or programs that interrupt access to transaction logs.

To delay Manager activities

Use the BOOTDELAYMINUTES parameter on a Windows system to delay the activities that Manager performs when it starts, such as executing parameters. For example, BOOTDELAYMINUTES can be used to delay AUTOSTART parameters until database services are started.

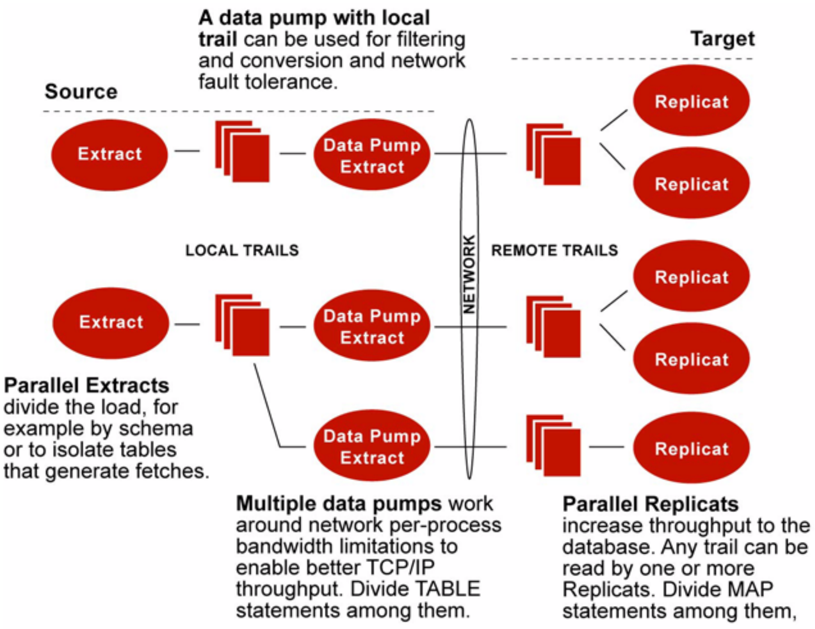
Balancing the load across parallel process groups

The most basic thing you can do to improve GoldenGate's performance is to divide a large number of tables among parallel processes and trails. For example, you can divide the load by schema.

Because each GoldenGate component — Extract, data pump, trail, Replicat — is an independent module, you can combine them in ways that suit your needs. You can use multiple trails and parallel Extract and Replicat processes (with or without data pumps) to improve performance and to isolate the processing of specific data. This configuration helps to minimize target latency when the transaction volume is high or when operations on certain tables cause bottlenecks.

Figure 6 shows some of the ways that you can GoldenGate to improve throughput speed and overcome network bandwidth issues.

Figure 6 Load-balancing configurations that improve performance


A data pump with local trail can be used for filtering and conversion and network fault tolerance.
Target
Source
Extract
Data Pump Extract
Replicat
Replicat
LOCAL TRAILS
NETWORK
REMOTE TRAILS
Replicat
Extract
Data Pump Extract
Replicat
Parallel Extracts divide the load, for example by schema or to isolate tables that generate fetches.
Data Pump Extract
Replicat
Multiple data pumps work around network per-process bandwidth limitations to enable better TCP/IP throughput. Divide TABLE statements among them.
Parallel Replicats increase throughput to the database. Any trail can be read by one or more Replicats. Divide MAP statements among them,

Considerations for using parallel process groups

Number of groups

The number of concurrent Extract and Replicat process groups that can run on a system depends on how much system memory is available. 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 GGSCI command interface fully supports up to 300 concurrent Extract and Replicat groups per instance of GoldenGate Manager. At the supported level, all groups can be controlled and viewed in full with GGSCI commands such as the INFO and STATUS commands. Beyond the supported level, group information is not displayed and errors can occur. GoldenGate recommends keeping the number of Extract and Replicat groups (combined) at 300 or below in order to manage your environment effectively.

NOTE When creating the groups, keep tables that have relational constraints to each other in the same group.

Isolating processing-intensive tables

Use dedicated process groups for certain kinds of tables that tend to interfere with normal processing and cause latency to build on the target. For example:

- Extract fetches data for large-object types, for columns that are not logged to the transaction log, and for data that must be fetched because of parameter specifications or SQL procedures. When data must be fetched from the database, it affects the performance of Extract because the data must be fetched using native calls. You can get fetch statistics from the STATS EXTRACT command if you include the STATOPTIONS REPORTFETCH parameter in the Extract parameter file.
- The Replicat process is most often a source of performance bottlenecks because, in its normal mode of operation, it is a single-threaded process that applies operations one at a time by using regular SQL. Replicat takes longer to process tables that have large or long-running transactions, heavy volume, a very large number of columns that change, and LOB data.

NOTE You can use the Logdump TRANSHIST command to view the size of the transactions that are contained in a trail file. See Chapter 4.

NOTE Depending on the types of transactions your applications generate, you might be able to gain processing efficiencies by using BATCHSQL instead of the normal Replicat mode. See the *GoldenGate for Windows and UNIX Reference Guide*.

Using parallel Replicat groups on a target system

In most cases, only a single instance of Extract is required to extract and transmit data to the target system, even for very large data loads. Typically, extracting and moving data to the target occurs near real-time, with backlogs typically occurring during the apply process on the target. Because Replicat must reproduce the source operations by constructing SQL, its I/O activity tends to be random access, as opposed to Extract I/O, which is serial, blocked, and buffered. To reduce this inherent latency, you can use multiple Replicat processes.

Although it is possible for multiple Replicat processes to read a single trail (no more than three of them to avoid disk contention) the best practice is to pair each Replicat with its own trail and corresponding Extract process.

When using parallel Replicats, configure each one to process a different portion of the overall data. To ensure data integrity, tables with referential integrity to one another must be processed through the same trail and associated Replicat group. For example, process table1 and table2 — and any child tables — through one trail-group pair, and process table3 and table4 and their child tables through another.

- Refer to the *GoldenGate for Windows and UNIX Reference Guide* for complete command and parameter syntax.
- For detailed instructions on configuring change synchronization, see the *GoldenGate for Windows and UNIX Administrator Guide*.

To create the Extract group

NOTE If needed, one or more Extract data-pumps can be added to this configuration.

1. On the source, use the ADD EXTRACT command to create an Extract group, specifying TRANLOG as the data source.

```
ADD EXTRACT <group>, TRANLOG, BEGIN <time> [, THREADS <n>]
```

2. On the source, use the ADD RMTTRAIL command to specify the full path name of a trail to be created by GoldenGate on the target system.

```
ADD RMTTRAIL <trail_1>, EXTRACT <group>
```

   ❍ Repeat this step to create a trail for each Replicat group that you will be using. For the EXTRACT argument of the command, specify the same Extract group.

```
ADD RMTTRAIL <trail_2>, EXTRACT <group>
```

3. On the source, use the EDIT PARAMS command to create an Extract parameter file that contains the following required parameters plus any others that are appropriate for your database environment.

```
EXTRACT <group>
[SOURCEDB <dsn>,][USERID <user>[, PASSWORD <pw>]]
RMTHOST <hostname>, MGRPORT <portnumber>
RMTTRAIL <trail_1>
TABLE <owner>.<table_group_1>;
RMTTRAIL <trail_2>
TABLE <owner>.<table_group_2>;
```

   ❍ Divide the source tables among different TABLE parameters, as many as you need. This example shows two.
   ❍ Link each TABLE statement to a different trail that is specified with the Extract parameter RMTTRAIL.
   ❍ If needed, you link multiple TABLE statements to any given trail.

To create the Replicat groups

4. On the target, create a Replicat checkpoint table. This is a best practice. For instructions, see the *GoldenGate for Windows and UNIX Administrator Guide*. All Replicat groups can use the same checkpoint table.

5. On the target, use the ADD REPLICAT command to create a Replicat group for each trail that you created. Use the EXTTRAIL argument of that command to link the group to a trail.

```
ADD REPLICAT <group_1>, EXTTRAIL <trail_1>
ADD REPLICAT <group_2>, EXTTRAIL <trail_2>
```

6. On the target, use the EDIT PARAMS command to create a Replicat parameter file for each Replicat group. In each file, use the following required parameters plus any others that are appropriatge for your database environment.

   Replicat group 1

   ```
   REPLICAT <group_1>
   {SOURCEDEFS <full_pathname>} | ASSUMETARGETDEFS
   [TARGETDB <dsn>,] [USERID <user id>[, PASSWORD <pw>]]
   MAP <owner>.<table_group_1>, TARGET <owner>.<table_group_1>;
   ```

   Replicat group 2

   ```
   REPLICAT <group_2>
   {SOURCEDEFS <full_pathname>} | ASSUMETARGETDEFS
   [TARGETDB <dsn>,] [USERID <user id>[, PASSWORD <pw>]]
   MAP <owner>.<table_group_2>, TARGET <owner>.<table_group_2>;
   ```

   - You can use any number of MAP statements for any given Replicat group. All MAP statements for a given Replicat group must specify the same objects that are contained in the trail that is linked to the group.

7. In the Manager parameter file on the target system, use the PURGEOLDEXTRACTS parameter to control the purging of files from the trails.

NOTE You can take this configuration a step further by dividing the rows of an individual table among multiple Replicat processes. This might be necessary, for example, when table1 receives one million inserts and updates per day, while the rest of the tables receive 100,000 or less. See "Splitting large tables into row ranges across process groups" on page 27.

Using parallel Extract groups with parallel Replicat groups

To ensure good performance on the source, you might need to use two or more parallel Extract processes in conjunction with parallel Replicat processes on the target. For example, candidates for a dedicated Extract are tables that generate fetches (generally those containing LOBs and columns that are not logged to the transaction log) and tables that generate long transactions or that trigger SQL procedures or user exits.

Multiple Extract groups extract and transmit in parallel to their own trails. Each trail is read by a dedicated Replicat group. Configure each Extract-Replicat pair to process a different portion of the overall data by assigning different tables to each one. To ensure data integrity, tables with referential integrity to one another must be processed through the same Extract-Replicat pair. For example, process table1 and table2 — and any child tables — through one pair, and process table3 and table4 and their child tables through another.

- Refer to the *GoldenGate for Windows and UNIX Reference Guide* for complete command and parameter syntax.
- For detailed instructions on configuring change synchronization, see *GoldenGate for Windows and UNIX Administrator Guide.*

To create the Extract groups

> **NOTE** If needed, an Extract data-pump can be added to each Extract-Replicat pair.

1. On the source, use the ADD EXTRACT command to create the Extract groups, specifying TRANLOG as the data source.

```
ADD EXTRACT <group_1>, TRANLOG, BEGIN <time> [, THREADS <n>]
ADD EXTRACT <group_2>, TRANLOG, BEGIN <time> [, THREADS <n>]
```

2. On the source, use the ADD RMTTRAIL command to create a trail for each Extract group.

```
ADD RMTTRAIL <remote_trail_1>, EXTRACT <group_1>
ADD RMTTRAIL <remote_trail_2>, EXTRACT <group_2>
```

3. On the source, use the EDIT PARAMS command to create an Extract parameter file for each Extract group. Use the following required parameters plus any others that are appropriate for your database environment.

Extract group 1

```
EXTRACT <group_1>
[SOURCEDB <dsn>,][USERID <user>[, PASSWORD <pw>]]
RMTHOST <hostname>, MGRPORT <portnumber>
RMTTRAIL <remote_trail_1>
TABLE <owner>.<table_group_1>;
```

Extract group 2

```
EXTRACT <group_2>
[SOURCEDB <dsn>,][USERID <user>[, PASSWORD <pw>]]
RMTHOST <hostname>, MGRPORT <portnumber>
RMTTRAIL <remote_trail_2>
TABLE <owner>.<table_group_2>;
```

- Divide the source tables among the Extract groups by using the TABLE parameter in each parameter file. This example shows two Extract parameter files.
- You can use any number of TABLE statements for any given Extract group.
- Use the Extract parameter RMTTRAIL to link each Extract group to a different trail.

To create the Replicat groups

4. On the target, create a Replicat checkpoint table. This is a best practice. For instructions, see the *GoldenGate for Windows and UNIX Administrator Guide*. All Replicat groups can use the same checkpoint table.

5. On the target, use the ADD REPLICAT command to create a Replicat group for each trail. Use the EXTTRAIL argument of that command to link the group to the trail.

```
ADD REPLICAT <group_1>, EXTTRAIL <remote_trail_1>, BEGIN <time>
ADD REPLICAT <group_2>, EXTTRAIL <remote_trail_2>, BEGIN <time>
```

6. On the target, use the EDIT PARAMS command to create a Replicat parameter file for each Replicat group. In each file, use the following required parameters plus any others that are appropriatge for your database environment.

Replicat group 1

```
REPLICAT <group_1>
{SOURCEDEFS <full_pathname>} | ASSUMETARGETDEFS
[TARGETDB <dsn>,] [USERID <user id>[, PASSWORD <pw>]]
MAP <owner>.<table_group_1>, TARGET <owner>.<table_group_1>;
```

Replicat group 2

```
REPLICAT <group_2>
{SOURCEDEFS <full_pathname>} | ASSUMETARGETDEFS
[TARGETDB <dsn>,] [USERID <user id>[, PASSWORD <pw>]]
MAP <owner>.<table_group_2>, TARGET <owner>.<table_group_2>;
```

- You can use any number of MAP statements for any given Replicat group. All MAP statements for a given Replicat group must specify the same objects that are contained in the trail that is linked to the group.

7. In the Manager parameter files on the source system and the target system, use the PURGEOLDEXTRACTS parameter to control the purging of files from the trails.

NOTE You can take this configuration a step further by dividing the rows of an individual table among multiple Extract or Replicat processes. This might be necessary, for example, when table1 receives one million inserts and updates per day, while the rest of the tables receive 100,000 or less. See "Splitting large tables into row ranges across process groups".

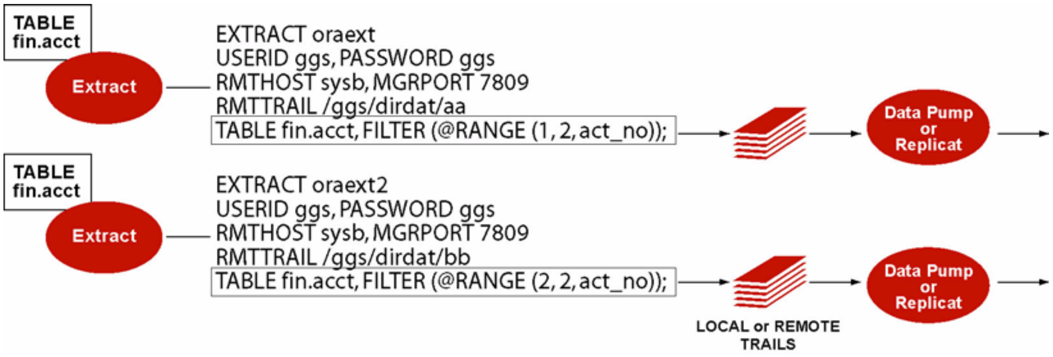
Splitting large tables into row ranges across process groups

Use the @RANGE function to divide the rows of any table across two or more GoldenGate processes. It can be used to increase the throughput of large and heavily accessed tables and also can be used to divide data into sets for distribution to different destinations. Specify each range in a FILTER clause in a TABLE or MAP statement.

@RANGE is safe and scalable. It preserves data integrity by guaranteeing that the same row will always be processed by the same process group.

It might be more efficient to use Extract to calculate the ranges than to use Replicat. To calculate ranges, Replicat must filter through the entire trail to find data that meets the range specification. However, your business case should determine where this filtering is performed.

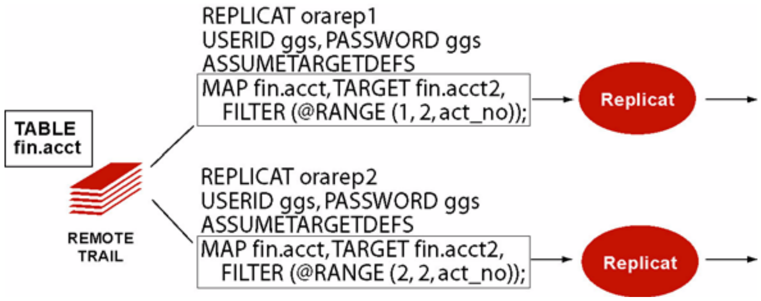
Figure 7 Dividing rows of a table between two Extract groups



```
EXTRACT oraext
USERID ggs, PASSWORD ggs
RMTHOST sysb, MGRPORT 7809
RMTTRAIL /ggs/dirdat/aa
TABLE fin.acct, FILTER (@RANGE (1, 2, act_no));
```

```
EXTRACT oraext2
USERID ggs, PASSWORD ggs
RMTHOST sysb, MGRPORT 7809
RMTTRAIL /ggs/dirdat/bb
TABLE fin.acct, FILTER (@RANGE (2, 2, act_no));
```

Figure 8 Dividing rows of a table between two Replicat groups



```
REPLICAT orarep1
USERID ggs, PASSWORD ggs
ASSUMETARGETDEFS
MAP fin.acct, TARGET fin.acct2,
  FILTER (@RANGE (1, 2, act_no));
```

```
REPLICAT orarep2
USERID ggs, PASSWORD ggs
ASSUMETARGETDEFS
MAP fin.acct, TARGET fin.acct2,
  FILTER (@RANGE (2, 2, act_no));
```

See the *GoldenGate for Windows and UNIX Administrator Guide* for more information about using parallel processing groups. Also see the GoldenGate installation guide for your database to see if there are any limitations that apply to how GoldenGate can be configured in your environment. For example, verify that the database can support multiple GoldenGate processes.

Tuning a data pump for pass-through efficiency

If a data pump is only being used to pass data from one process to another, without performing filtering, mapping, or conversion, configure it to run in pass-through mode. In this mode, the Extract process bypasses its normal queries for table definitions from the database or a definitions file and can even be used on a system that does not contain a database. To use pass-through mode, include the PASSTHRU parameter in the parameter file of the data pump.

Configuring GoldenGate to use the network efficiently

Inefficiencies in the transfer of data across the network can cause lag in the Extract process and latency on the target. If not corrected, it can eventually cause process failures.

When you first start a new GoldenGate configuration:

1. Establish benchmarks for what you consider to be acceptable lag and throughput volume for Extract and for Replicat. Keep in mind that Extract will normally be faster than Replicat because of the kind of tasks that each one performs. Over time you will know whether the difference is normal or one that requires tuning or troubleshooting.
2. Set a regular schedule to monitor those processes for lag and volume, as compared to the benchmarks. Look for lag that remains constant or is growing, as opposed to occasional spikes. Continuous, excess lag indicates a bottleneck somewhere in the GoldenGate configuration. It is a critical first indicator that GoldenGate needs tuning or that there is an error condition.

To view volume statistics

Syntax

```
STATS {EXTRACT | REPLICAT | ER} {<group | wildcard>}
[TABLE {<name | wildcard>}]
```

To view lag statistics

Use either the LAG or SEND command in GGSCI.

Syntax

```
LAG {EXTRACT | REPLICAT | ER} {<group | wildcard>}
```

Or...

Syntax

```
SEND {EXTRACT | REPLICAT} {<group | wildcard>}, GETLAG
```

To detect a network bottleneck

1. Issue the following command to view the ten most recent Extract checkpoints. If you are using a data-pump Extract on the source system, issue the command for the primary Extract and also for the data pump.

   ```
   INFO EXTRACT <group>, SHOWCH 10
   ```

2. Look for the Write Checkpoint statistic. This is the place where Extract is writing to the trail.

   ```
   Write Checkpoint #1

       GGS Log Trail

       Current Checkpoint (current write position):

           Sequence #: 2
           RBA: 2142224
           Timestamp: 2006-06-09 14:16:50.567638
           Extract Trail: ./dirdat/eh
   ```

3. For both the primary Extract and data pump:
   - Determine whether there are more than one or two checkpoints. There can be up to ten.

- Find the Write Checkpoint <n> heading that has the highest increment number (for example, Write Checkpoint #8) and make a note of the Sequence, RBA, and Timestamp values. This is the most recent checkpoint.

4. Refer to the information that you noted, and make this validation:
   - Is the primary Extract generating a series of checkpoints, or just the initial checkpoint?
   - If a data pump is in use, is it generating a series of checkpoints, or just one?

5. Issue INFO EXTRACT for the primary and data pump Extract processes again.
   - Has the most recent write checkpoint increased? Look at the most recent Sequence, RBA, and Timestamp values to see if their values incremented forward since the previous INFO EXTRACT command.

6. Issue the following command to view the status of the Replicat process.

   ```
   SEND REPLICAT <group>, STATUS
   ```

   - The status indicates whether Replicat is delaying (waiting for data to process), processing data, or at the end of the trail (EOF).

7. There is a network bottleneck if:

   You are only using a primary Extract and its write checkpoint is not increasing or is increasing too slowly. Because this Extract process is responsible for sending data across the network, it will eventually run out of memory to contain the backlog of extracted data and abend.

   Or...

   You are using a data pump, and its write checkpoint is not increasing, but the write checkpoint of the primary Extract is increasing. In this case, the primary Extract can write to its local trail, but the data pump cannot write to the remote trail. The data pump will abend when it runs out of memory to contain the backlog of extracted data. The primary Extract will run until it reaches the last file in the trail sequence and will abend because it cannot make a checkpoint.

   And...

   The status of Replicat is either in delay mode or at the end of the trail file. Even when there is a network outage, Replicat will process in a normal manner until it applies all of the remaining data from the trail to the target. Eventually, it will report that it reached the end of the trail file.

To get more network bandwidth

Use parallel data pumps to overcome any per-process bandwidth limitations of your network. Data pumps also remove TCP/IP responsibilities from the primary Extract, and their local trails provide fault tolerance. For more information, see the *GoldenGate for Windows and UNIX Administrator Guide.*

To reduce bandwidth requirements

Use the compression options of the RMTHOST parameter to compress data before it is sent across the network. Weigh the benefits of compression against the CPU resources that are required to perform the compression.

To increase the TCP/IP packet size

Use the TCPBUFSIZE option of the RMTHOST parameter to increase the size of the TCP socket buffer that Extract maintains. By increasing the size of the buffer, you can send larger packets to the target system.

Eliminating disk I/O bottlenecks

I/O activity can cause throughput bottlenecks for both Extract and Replicat.

- A regular Extract generates disk writes to a trail and disk reads from a data source.
- A data pump and Replicat generate disk reads from a local trail.
- Each process writes a recovery checkpoint to its checkpoint file on a regular schedule.

To improve I/O within the system configuration

Look for I/O waits on the disk subsystems that contain the trail files. Put the trails on the fastest disk controller possible.

Check the RAID configuration. Because GoldenGate writes data sequentially, RAID 0+1 (striping and mirroring) is a better choice than RAID 5, which uses checksums that slow down I/O and are not necessary for these types of files.

To improve I/O within the GoldenGate configuration

Try increasing the values of the following parameters:

Use the CHECKPOINTSECS parameter to control how often Extract and Replicat make their routine checkpoints.

Use the GROUPTRANSOPS parameter to control the number of SQL operations that are contained in a Replicat transaction when operating in its normal mode. Increasing the number of operations in a Replicat transaction improves GoldenGate's performance by:

- Reducing the number of transactions executed by Replicat.
- Reducing I/O activity to the checkpoint file and the checkpoint table, if used. Replicat issues a checkpoint whenever it applies a transaction to the target, in addition to its scheduled checkpoints.

Use the EOFDELAY or EOFDELAYCSECS parameter to control how often Extract, a data pump, or Replicat checks for new data after it has reached the end of the current data in its data source. You can reduce the system I/O overhead of these reads by increasing the value of this parameter.

NOTE Increasing the values of these parameters will improve performance, but it also increases the amount of data that must be reprocessed if the process fails. This has an effect on overall latency between source and target. Some testing will help you determine the optimal balance between recovery and performance.

Optimizing data filtering and conversion

Data filtering and data conversion both add overhead, and these activities are sometimes prone to configuration errors. The following are best practices for minimizing the impact of this overhead on the other processes on the system.

- Avoid using the primary Extract to filter and convert data. Keep it dedicated to data capture. It will perform better and is less vulnerable to any process failures that result from those activities.
- Use Replicat or a data-pump to perform filtering and conversion. Consider any of the following configurations:
  - Use a data pump on the source if the system can tolerate the overhead. This configuration works well when there is a high volume of data to be filtered, because it uses less network bandwidth. Only filtered data gets sent to the target, which also can help with security considerations.
  - Use a data pump on an intermediate system. This configuration keeps the source and target systems free of the overhead, but uses more network bandwidth because unfiltered data is sent from the source.
  - Use a data pump or Replicat on the target if the system can tolerate the overhead, and if there is adequate network bandwidth for sending large amounts of unfiltered data.
- Consider using parallel data pumps or parallel Replicat processes to balance the load of filtering and conversion. Parallel processes work well when sending data to different target systems, because you can dedicate one or more pumps to each target. Keep tables that have relational constraints to each other in the same process group.
- If you have limited system resources, a least-best option is to divide the filtering and conversion work between Extract and Replicat.

For more information about configuring processes, see the *GoldenGate for Windows and UNIX Administrator Guide*.

Applying similar SQL statements in arrays

Use the BATCHSQL parameter to increase the performance of Replicat. BATCHSQL causes Replicat to organize similar SQL statements into arrays and apply them at an accelerated rate. In its normal mode, Replicat applies one SQL statement at a time.

When Replicat is in BATCHSQL mode, smaller row changes will show a higher gain in performance than larger row changes. At 100 bytes of data per row change, BATCHSQL has been known to improve Replicat's performance by up to 300 percent, but actual performance benefits will vary, depending on the mix of operations. At around 5,000 bytes of data per row change, the benefits of using BATCHSQL diminish.

Preventing full table scans in the absence of keys

If a target table does not have a primary key, a unique key, or a unique index, Replicat uses all of the columns to build its WHERE clause. This is, essentially, a full table scan. Use a KEYCOLS clause in the TABLE and MAP statements to specify one or more columns for Replicat to use as a key. The columns must contain unique values. On the target system, create an index on those columns.

```
TABLE <source_table>, KEYCOLS (<column> [, <column>, <column>, ...]);
```

and...

```
MAP <source_table>, TARGET <target_table>, KEYCOLS (<column>
[, <column>, <column>, ...]);
```

Example

```
TABLE hr.emp, KEYCOLS (FIRST_NAME, LAST_NAME, DOB, ID_NO);
MAP hr.emp, TARGET hr.emp2, KEYCOLS (FIRST_NAME, LAST_NAME, DOB, ID_NO);
```

Splitting large transactions

If the target database cannot handle large transactions from the source database, you can split them into a series of smaller ones by using the Replicat parameter MAXTRANSOPS.

Tuning Replicat transactions

Replicat uses regular SQL, so its performance to some degree depends on the performance of the target database and the type of SQL that is being applied (inserts, versus updates or deletes). However, you can take the following steps to maximize Replicat efficiency.

Adjust open cursors

The Replicat process maintains cursors for cached SQL statements and for SQLEXEC operations. Without enough cursors, Replicat must age more statements. By default, Replicat maintains as many cursors as allowed by the MAXSQLSTATEMENTS 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.

Improve update speed

Excessive block fragmentation causes Replicat to apply SQL statements at a slower than normal speed. Reorganize heavily fragmented tables, and then stop and start Replicat to register the new object ID.

Ensure effective execution plans

Stale database statistics can reduce the performance of Replicat as it adds rows to tables. Generate new statistics to ensure the use of appropriate execution plans.

Set a Replicat transaction timeout

Use the TRANSACTIONTIMEOUT parameter to prevent an uncommitted Replicat target transaction from holding locks on the target database and consuming its resources unnecessarily. You can change the value of this parameter so that Replicat can work within existing application timeouts and other database requirements on the target.

TRANSACTIONTIMEOUT limits the amount of time that Replicat will hold a target transaction open if it has not received the end-of-transaction record for the last source transaction in that transaction. By default, Replicat groups multiple source transactions into one target transaction to improve performance, but it will not commit a partial source transaction and will wait indefinitely for that last record. The Replicat parameter GROUPTRANSOPS controls the minimum size of a grouped target transaction.

The following events could last long enough to trigger TRANSACTIONTIMEOUT:

- Network problems prevent trail data from being delivered to the target system.
- Running out of disk space on any system, preventing trail data from being written.
- Collector abends (a rare event).
- Extract abends or is terminated in the middle of writing records for a transaction.
- An Extract data pump abends or is terminated.
- There is a source system failure, such as a power outage or system crash.

CHAPTER 3

Troubleshooting GoldenGate

Overview of GoldenGate troubleshooting

One of the challenges of solving problems in a replication environment is in deciding which component of the environment is at the root of the problem:

- the system or network?
- the database or applications?
- the GoldenGate installation?
- a specific GoldenGate process?
- the way that GoldenGate is configured?
- SQL or procedures?

The GoldenGate troubleshooting documentation provides help with many common issues that can arise during GoldenGate startup and the processing of data. You can find help for other known issues and frequently asked questions by going to the GoldenGate Technical Support website at http://support.goldengate.com/ and viewing the Knowledge Base. Troubleshooting tools described in this chapter are documented in Chapter 1.

Important: The more invasive of the troubleshooting solutions, such as those affecting data records and processing continuity, should be performed by individuals who have authority to resolve problems and who understand the GoldenGate components, configuration structure, and processing methodology. Less experienced users should undertake such solutions in conjunction with GoldenGate Technical Support staff or a more experienced user.

Troubleshooting example

Here is a simple example of how you can use GGSCI commands and the error log to determine why a process (in this case Extract group ggext) failed.

1. The STATUS EXTRACT command returns the following:

   ```
   GGSCI (sysa) 20> status extract ggext
   EXTRACT GGEXT: ABENDED
   ```

2. Viewing the error log with VIEW GGSEVT shows the following:

   ```
   GGSCI (sysa) 22> view ggsevt
   2006-04-27 16:38:14  GGS INFO    399  GoldenGate Command Interpreter for
   Oracle:  GGSCI command (admin): start extract ggext.
   2006-04-27 16:38:15  GGS INFO     301  GoldenGate Manager for Oracle,
   mgr.prm:  Command received from GGSCI on host 100.000.000.000 (START
   EXTRACT GGEXT).
   2006-04-27 16:38:15  GGS INFO     310  GoldenGate Capture for Oracle,
   ggext.prm:  EXTRACT GGEXT starting.
   2006-04-27 16:38:17  GGS ERROR    501  GoldenGate Capture for Oracle,
   ggext.prm:  Extract read, error 13 (Permission denied) opening redo log
   /newrdbms/oracle/ora920/arch/0001_0000000758.arc for sequence 758.
   2006-04-27 16:38:17  GGS ERROR    190  GoldenGate Capture for Oracle,
   ggext.prm:  PROCESS ABENDING.
   ```

   The error message 501 indicates the Extract user does not have permission to read the redo logs.

Solution: Contact the system administrator to grant the user running the extract program read permission on those files. Then do the following:

1. Stop Manager.
2. Exit GGSCI.
3. Exit the terminal session.
4. Start the processes again from a new session.

Troubleshooting process startup failures

When a process fails, view the process report and the error log. Check for the following conditions.

Configuration problems that cause startup failure

✔ Was the correct GoldenGate build installed?

- GoldenGate provides a unique build of the software for each type and version of the operating systems and databases that it supports. The build name contains the operating system version, database version, GoldenGate release number, and GoldenGate build number, as shown in the following example for Oracle on AIX:

  ```
  aix51_ora92_v8020_023.tar.gz
  aix51_ora817_v8020_023.tar.gz
  aix52_ora102_v8020_023.tar.gz
  aix53_ora92_v8020_023.tar.gz
  aix53_ora101_v8020_023.tar.gz
  aix53_ora102_v8020_023.tar.gz
  ```

- To find out the GoldenGate build and version that is installed, go to the GoldenGate home directory and issue the following command from the operating system:

  ```
  ggsci -v
  ```

✔ Did you use the CHECKPARAMS parameter and forget to remove it?

- This parameter causes the process to stop after it verifies parameter syntax.
- View the process report for syntax errors, fix them, and then remove CHECKPARAMS.

✔ Is the database running? Is Manager running?

- Both must be running before starting process groups.

✔ Does the process group that you are trying to start exist?

- Use INFO ALL to view all processes and groups on the system. Maybe there was a spelling error when the group was created or when the START command was issued.

✔ Does a trail or extract file exist?

- If not, Extract cannot write its initial checkpoint, and Replicat has no data source to read.
- Use INFO EXTRACT <group> or INFO REPLICAT <group> with the DETAIL option to see if a trail or file exists.

✔ Are parameter files in the correct place?

- Check for the GLOBALS file (if being used) in the root GoldenGate installation directory. It should not have a file extension.
- Check for the Manager parameter file and any other parameter files in the GoldenGate dirprm sub-directory. The Manager file must be named MGR.prm, and all other parameter files should have the name of the associated process group.
- If a parameter file is not there, did you store it elsewhere? Use the INFO EXTRACT <group>, DETAIL command to see where else it might be. To store a parameter file elsewhere (not recommended), use the PARAMS argument with ADD EXTRACT or ALTER EXTRACT.

✔ Is the parameter file accessible?

- Check the permissions for the system user assigned to the process. Grant read permission, if needed.

✔ Are all required parameters present?

- Requirements vary by database and configuration. Review the requirements for your database in the *GoldenGate for Windows and UNIX Administrator Guide* and in the GoldenGate installation guide that is specific to the database type.

✔ Is parameter syntax correct?

- GoldenGate reports syntax problems in the process report, usually as a “bad parameter” error.

- Use the CHECKPARAMS parameter to verify syntax. It verifies syntax when the process starts, writes results to the report file, and then stops the process. After using CHECKPARAMS, remove it or comment it out; otherwise the process will stop again.
- Common syntax errors:
  - TABLE or MAP not terminated with a semi-colon.
  - Comma not followed by a space.
  - Missing commas, quotes, or parentheses in nested clauses, such as a COLMAP.
- See the *GoldenGate for Windows and UNIX Reference Guide* for complete parameter syntax information.

✔ Are parameters listed in logical order?

- Parameters are processed in the order listed; some must precede others.
- Parameter ordering examples:

  RMTHOST must precede RMTTRAIL: A host name must be qualified before a file name can be qualified.

  SOURCEDB or TARGETDB and USERID must precede TABLE: GoldenGate must log into the database before it can process table information.

  TABLE or MAP must be listed after global and specific parameters that apply to it.

Operating system errors that cause startup failure

✔ Does the error report a missing UNIX library?

- Issue the env command and make sure LD_LIBRARY_PATH and PATH are correct.
- Set the paths correctly in the .profile startup file in the GoldenGate home directory.

✔ Does the error request the function stack to be increased?

- The memory that stores arguments supplied to and from GoldenGate functions has been exceeded.
- To increase the memory allocated, use the FUNCTIONSTACKSIZE parameter. This parameter can adversely affect GoldenGate's performance.

✔ Are there file-access errors?

- Extract and Replicat users require the following permissions:
  - read and write for files in the GoldenGate directory.
  - read for the transaction and archive logs.

✔ Is the error “Bad parameter: Group name invalid”?

- The process cannot open the checkpoint file.
- Issue INFO * in GGSCI to view the group name, then issue VIEW PARAMS <group>. Make sure the group name from INFO * matches the one in the EXTRACT or REPLICAT parameter.
- Check the file permissions.

- On NonStop systems, look for access problems related to Safeguard and Enform reports.

Database errors that cause statup failure

- ✔ Is the database running?
- ✔ Was a database user created for the process and does it have the correct permissions?
    - Extract and Replicat require a database user with certain permissions. See the GoldenGate installation guide that is specific for your database type.
- ✔ If a data source name is required as part of the login, is it specified with SOURCEDB or TARGETDB?
- ✔ If database authentication is required, are the correct database user and password specified with the USERID and PASSWORD parameters?
    - Connect to the database through its native interface by using the same user and password that is contained in the parameter file of the failed process. The login that a process uses is specified with the USERID and PASSWORD parameters. Connecting from outside GoldenGate helps to identify whether the authentication itself is valid.
    - If you cannot connect from outside GoldenGate, check the database configuration to make certain the user and password are still valid. Perhaps a change was made to the user or password.
    - After you identify the correct login, update the USERID and PASSWORD parameters in the parameter file by using the EDIT PARAMS command. Stop and start the process when you are finished.
    - (Oracle) If database passwords change frequently, you can specify operating-system authentication with the / option of the USERID parameter.
- ✔ (Oracle) Are the ORACLE_SID and ORACLE_HOME system variables set to the correct instance name in the GoldenGate user profile?
    - To check Oracle environment variables, use:

      ```
      env | grep ORA
      ```

      This filters out all the other information that would be shown if you were to use the basic env command.

Troubleshooting slow startup

- ✔ Are wildcards being used for table names?
    - Make certain that the WILDCARDRESOLVE parameter is set to the default of DYNAMIC, so that tables satisfying wildcard definitions are resolved whenever the wildcard rule is satisfied, instead of all at once at startup.

✔ Is Extract operating in archived log only (ALO) mode?

- Did you create the group with the BEGIN NOW option of the ADD EXTRACT command? If an Extract is created that way and also configured for ALO mode, the Extract might not capture data immediately upon startup. 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 waits 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. This can happen in both regular and RAC database configurations.
- On RAC systems this effect is compounded by the fact that Extract needs to read data from all RAC instances before it can proceed. Extract will wait for Oracle to archive all of the online logs that correspond to the NOW time.
- To force capture to begin sooner, you can switch the logs manually with the following command:

```
ALTER SYSTEM SWITCH LOGFILE
```

Troubleshooting Extract problems

Extract freezes during recovery

If Extract abends due to an error or system failure, it might appear to be stalled when it is restarted. The cause of this is probably because Extract had to search a long way back into the transaction logs to find the beginning of a long-running transaction that was open at the time of the failure.

To find out whether a prolonged recovery is the cause of the delay, use the SEND EXTRACT command with the STATUS option. If one of the following messages is displayed on the Current status line, this indicates that recovery is in progress (or being completed) and that Extract is operating normally:

- In recovery[1] – Extract is recovering to its input checkpoint.
- In recovery[2] – Extract is recovering to its output checkpoint.
- Recovery complete – The recovery is finished, and normal processing will resume.

Extract is too slow

This section covers possible acute causes of slow extraction processing. For other things that affect the speed of extraction, and their solutions, see “Tuning GoldenGate’s Performance” on page 21.

✔ Is tracing enabled?

- Check the parameter file for TLTRACE and TRACE. Tracing slows down Extract.
- Disable those parameters unless required by a support case.

✔ Is there is a problem with the transfer of data across the network?

- See "To detect a network bottleneck" on page 29 for instructions that help you detect whether a network problem is affecting GoldenGate.
- Work with your network administrator to resolve network problems as soon as possible. If Extract lags too far behind the speed at which transaction logs are created, latency on the target will increase to an intolerable level, and Extract could eventually lose access to data that it still needs if the archives are moved off the system. Major network delays can also cause eventual process failure.
- If the network is functional, check to see if the per-process bandwidth has been used up for the Extract process. You might need to split the data load among some data pumps to gain more bandwidth. The use of data pumps also reduces the chance of process failures and data loss if the network fails intermittently. To configure data pumps, see the *GoldenGate for Windows and UNIX Administrator Guide*.

✔ Are there numerous columns in the tables?

- The more columns that are changed in a table, the longer it takes for Extract to process them. This is normal and cannot be avoided. You can isolate tables like this that cause slowdowns by using parallel processes. See page 22 for more information.

✔ Is Extract waiting for a transaction log?

- Extract waits if it cannot find the transaction log that it needs, which means it probably was archived off the system or was moved to a location that is unknown to GoldenGate. This can happen if there is high Extract lag and the archives are moved off the system, or if there are long-running open transactions, because Extract keeps a pointer to that log in the expectation of eventually receiving a commit operation.

  NOTE This could take hours or days of searching, and Extract will eventually abend if it cannot locate the log.

- To determine which log is needed, use the VIEW REPORT <group> command. Restore that log to the directory where it was before, and also restore subsequent ones until Extract is finished with them.
- (Oracle) If Oracle archives are not stored in Oracle's default location, use the Extract parameter ALTARCHIVELOGDEST <path name> to specify the alternate location.
- (Oracle) You can check for and manage open transactions against Oracle tables from within GoldenGate by using options of the SEND EXTRACT command. You can use the WARNLONGTRANS parameter to set a threshold after which a transaction is considered long-running. To check for open transactions from the database interface, query the V$Transaction table.
- If the transaction logs that GoldenGate requires are no longer available, resynchronize the source and target data. See "Resynchronizing an out-of-sync table" on page 53.

✔ See also "Extract stalls or pauses".

Extract stalls or pauses

✔ Is this an Oracle RAC system, and does Extract pause across all RAC nodes?

- On RAC, if an Extract thread reaches the end of a node's last archived log, and the next log is not immediately available, the entire Extract process will stop, even though other Extract threads have archive logs available. This is to maintain transactional integrity. ALO mode does not provide enough information for Extract to determine if there is other, older transaction data somewhere that must be processed before it processes the data captured by the other threads.

✔ Is this an Oracle RAC system, and did one of the instances fail?

- If one or more of the RAC instances fails, archive log generation will stop for those instances. In this situation, an Extract operating in ALO mode will stop. To allow Extract to continue, set the THREADOPTIONS paramter with the PROCESSTHREADS {SELECT | EXCEPT} option to skip processing on the failed nodes. This is an undocumented parameter. Please contact GoldenGate Technical Support before using this parameter.

Extract abends

✔ Is Extract unable to open the transaction logs?

- Check to see if the disk is full.
- Verify that the Extract user has read permissions on the log files.

  If you have to grant permissions, do the following afterward:

  1. Stop Manager.
  2. Exit GGSCI.
  3. Close the terminal session.
  4. Start processes again from a new session.

✔ Is an archived transaction log missing?

- Extract reads the archived logs when an online log is not available. If you started Extract after it was stopped for a long time, the records it needs to process probably were archived away.
- Restore the archive log that is specified in the error message in the report file, and also restore all subsequent logs.

NOTE Extract should not be stopped for a long time. If you must stop Extract, make certain that archives are available all the way back to the time that you stopped Extract. If Extract remains stopped for an exceptionally long time, it might be more practical to resynchronize the table(s) rather than wait for Extract to go through the archives and catch up.

✔ Can the source system communicate with the target?

- To check, use the ping shell command:

```
ping <host name>
```

The following shell command displays the network routing table:

```
netstat -r
```

✔ Does Extract return “connection-refused” errors?

- A typical TCP/IP error is “4127 connection refused.” It indicates that the target Manager or Collector process is not running, or that Extract is pointing to the wrong TCP/IP address or Manager port number.
- On the target, use INFO MGR in GGSCI to view the port number that is specified for the target Manager.
- On the source, check the Extract parameter RMTHOST for the following:
- Verify that MGRPORT is the same as the port shown by INFO MGR.
- If a host name was given for RMTHOST, make sure the domain name server (DNS) can resolve it. If an IP address was used, verify that it is correct. To verify an IP address, use the following commands from the OS command shell:

  UNIX:

  ```
  ifconfig
  ```

  Windows:

  ```
  ipconfig
  ```

- Connection errors can indicate Collector security violations, a full file system, or errors relating to the system or to the GoldenGate configuration. Use VIEW GGSEVT to check for those errors.

✔ Does Extract return the error “No Dynamic ports available”?

- The target Manager was unable to obtain a port on which to communicate with the source Manager. Manager looks for a port in the list specified with the DYNAMICPORTLIST parameter in its parameter file or, if that parameter is not in use, the next available port higher than the one it is running on, as specified with the PORT parameter.
- If you are using DYNAMICPORTLIST, it is possible that there are not enough numbers to use, or that the numbers are being used by an active or orphan process. Try adding some port numbers to the list. Or, contact your systems administrator to find out which ports can be used.
- If you are using GoldenGate Director to connect to the target server, you will need to increase the number of ports in your dynamic port list, because Director uses ports from the dynamic port list.

✔ Are you trying to use a data pump in pass-through mode?

- If you are using the PASSTHRU parameter for the data pump, but are trying to use data filtering, conversion, or other manipulation, this configuration will not work. In pass-through mode, the source and target table names and structures must be identical, and no filtering or manipulation can be done to the data.
- Does the parameter file that includes PASSTHRU also include USERID or SOURCEDB parameters? If the system does not have a database, these parameters should not be used.

- Are you trying to combine pass-through for some tables with normal processing for other tables? For the tables being processed normally, the system must have a database. The database login parameters of SOURCEDB and USERID must be used, and a source definitions file must be used if filtering is to be performed. A target definitions file must be used if column mapping or conversion is to be done.

✔ Is the error for a Sybase source that states “Failed to reserve LTM context”?

- Check to see what other processes are using the LTM. Extract uses the LTM to read the transaction log, and only one process at a time can reserve a context. Therefore, no other Extract or RepServer processes can be run against the same database. Errors similar to the following in the error log indicate that some other process is running.

```
2005-01-18 22:54:01  GGS INFO     631  SYBLTM - Server message:.
2005-01-18 22:54:01  GGS INFO    631  number(9121) severity(16) state(2)
line(1).
2005-01-18 22:54:01  GGS INFO     631   Server name: SysA.
2005-01-18 22:54:01  GGS INFO     631   The log transfer context for the
current database is already reserved by SQL Server process 16.  The log
transfer context for the current database is not reserved.
2005-01-18 22:54:01  GGS ERROR    630  Failed to reserve LTM context
during log reader initialization.
2005-01-18 22:54:01  GGS ERROR    190  PROCESS ABENDING.
```

✔ Is the database DB2 For LUW?

- Is the database configured to retain the logs? Either USEREXIT or LOGRETAIN must be enabled.
- Is the DB2 parameter OVERFLOWLOGPATH set to the archive log directory? This enables GoldenGate to get archived data stored in a non-default location.
- Are you using wildcards to specify tables? Check to see if DATA CAPTURE CHANGES was specified for all of the tables that satisfy the wildcard conditions. If not, use the TRANLOGOPTIONS parameter with the IGNOREDATACAPTURECHANGES option. This configuration directs Extract to ignore tables for which change capture is not enabled. A warning will be issued to the error log that those tables were skipped.
- Have columns been added with the ALTER command to a table that is already in the DB2 LUW configuration? And, have any of those columns been defined as key columns? If so, check to see if there were updates on those columns. A primary key update requires a before image for GoldenGate to compare to the after image, but the DB2 API used by GoldenGate does not provide the before image for the added columns. This will cause Extract to abend. Columns added to existing tables cannot be part of a key in the data definitions or as part of a KEYCOLS clause in the GoldenGate parameters TABLE or MAP.
- Is the error similar to the following example? If so, Extract does not have the correct permissions to read the catalog tables and log files.

```
[SC=-1224:SQL1224N A database agent could not be started to service a
request, or was terminated as a result of a database system shutdown or
a force command. SQL STATE 55032: The CONNECT statement is invalid,
because the database manager was stopped after this application was
started].
```

✔ Is the database DB2 For z/OS?

- Are the correct permissions set? The API (GoldenGate name of IFI IFCID 306) requires Extract to call the API in Supervisor State with Storage Key 0 and allocate a return area in Extended Common Storage Area (ECSA) memory subpool 241 with Storage Key 7. For more information about required privileges, see the installation guide for the DB2 for z/OS database.

✔ Does Extract return an error on LOB length when extracting from a SQL Server 2005 source?

```
Msg 7139, Level 16, State 1, Line 1
Length of LOB data (<n_bytes>) to be replicated exceeds configured
maximum 65536.
The statement has been terminated.
```

- This means that the data for LOB or VARCHAR (MAX) columns exceeds the SQL Server default size.
- To resolve this issue, extend the maximum data size with the 'max text repl size' parameter by using sp_configure. For example:

```
'max text repl size', '2147483647'
```

Use sp_configure to find the current default value of 'max text repl size'.

Troubleshooting Replicat problems

Replicat is too slow

✔ Is there consistent latency in the target data?

- Is the DEFERAPPLYINTERVAL being used in the Replicat parameter file? To find out, you can examine the parameter file or use the following GGSCI command and look for a status of "Waiting on deferred apply."

```
    SEND REPLICAT <group> STATUS
```

- This parameter causes an intentional, specific delay in the timing of Replicat's transactions. Replicat will always wait for the specified amount of time after reading data from the trail, before applying that data to the target table.
- If your target applications do not require this delay, you can remove DEFERAPPLYINTERVAL. You must stop Replicat and then start it again for the change to take effect.

✔ Does Replicat appear stalled?

- Issue the following command in GGSCI a few times to see whether or not the checkpoint is moving.

```
INFO REPLICAT <group>, SHOWCH
```

- Look for the Read checkpoint. If the number of Read checkpoints is not increasing, Replicat is stalled.

Do the following:

1. Note the trail name that Replicat is reading from, as shown with INFO REPLICAT.
2. Issue INFO EXTRACT <group>. Note the trail name that Extract is writing to.
3. Make certain that Replicat is reading the same trail that Extract is writing to. If not, issue ALTER REPLICAT <group>, EXTTRAIL <trail> to specify the correct trail.
4. Issue INFO RMTTRAIL * (or INFO EXTTRAIL * if local) and make sure the trail was created. Create the trail, if necessary, with the following command:

   ```
   ADD {EXTTRAIL | RMTTRAIL} <trail>, EXTRACT <group>.
   ```
5. Make certain that the RMTTRAIL or EXTTRAIL Extract parameter is also set to the correct trail name.

❍ If the checkpoint is moving, Replicat is processing data, and other issues are causing the slowdown. See “Tuning GoldenGate’s Performance” on page 21 for conditions that might be affecting Replicat’s performance.

✔ Is Replicat holding locks on target objects?

- Problems with the network, loss of disk space, process failures, and system failures all can interrupt the transfer of source operations to the trail. If Replicat cannot commit its current transaction because it is waiting for the rest of the last source transaction in the transaction group to arrive, the lock will be held indefinitely.
- If you continue to see that Replicat is holding locks, and you cannot resolve the reasons that are delaying source transactions, consider using the Replicat TRANSACTIONTIMEOUT parameter to allow Replicat to handle the condition.

Replicat abends

✔ Are there checkpoint table errors?

- Depending on the problem, these errors can appear as the following:

```
GGS ERROR 516 Extract read, No data found selecting position from
checkpoint table HR.REPCHECK for group REPORA, key 1181389907
(0x4669453).
```

Or...

```
2006-09-25 13:22:16 GGS ERROR 516 Extract read, Checkpoint table
HR.REPCHECK does not exist.  Please create the table or recreate the
REPORA group using the correct table.
```

- The checkpoint table may be corrupted, deleted, or missing a row. Do the following:

  1. Issue INFO REPLICAT <group> to verify that the checkpoint file on disk is intact. If the command fails, the file is missing or corrupt. If the file is intact, continue with these steps.
  2. If the table was deleted, recreate it with same schema and name as before. If the table is corrupt or missing a row, truncate it.
  3. From the GoldenGate installation directory, run the convchk utility, specifying the checkpoint table as shown:

     ```
     convchk <group name> <schema>.<table>
     ```

  4. Start Replicat.
  5. SELECT from the checkpoint table to verify that checkpoints are being added again.

  NOTE Running convchk causes the checkpoint file to become the master source for the group's checkpoints. When started, Replicat will populate the new or truncated checkpoint table with the data from the checkpoint file, in effect resynchronizing them.

✔ Is Replicat unable to locate a trail?

- Do the following:

  1. Issue the following command, and record the trail name to which Extract is writing.

     ```
     INFO EXTRACT <group>
     ```

  2. Issue the following command to see if the trail was created.

     ```
     INFO {RMTTRAIL | EXTTRAIL} *
     ```

  3. Create the trail, if necessary, with the following command.

     ```
     ADD {EXTTRAIL | RMTTRAIL} <trail>, EXTRACT <group>
     ```

  4. Issue the following command to link that trail to the Replicat process.

     ```
     ALTER REPLICAT <group>, EXTTRAIL <trail>
     ```

  5. Make certain that the RMTTRAIL or EXTTRAIL Extract parameter is also set to the correct trail name.

✔ Is Replicat unable to open a trail file?

- Verify that the Replicat user has read and write permissions on trail files.

✔ Are you using trail encryption?

- If you specified ENCRYPTTRAIL for Extract, make certain that DECRYPTTRAIL is specified for Replicat. Otherwise, Replicat will return GoldenGate error 160. The encrypted data will be interpreted as a bad record.

✔ Are Replicat's transactions large?

- If the target cannot handle large transactions, Replicat will abend with a memory error.

- Try using the MAXTRANSOPS parameter to split Replicat's transactions into smaller ones.

✔ Have source definitions changed?

- For example, were columns added to a source table?
- If so, did you create a new source-definitions file for that table? Use DEFGEN and then transfer the new file to the target.

✔ Are you replicating LOBs to a Sybase target?

- Are you moving a LOB to a Sybase CHAR, VARCHAR, BINARY, or VARBINARY column? If so, the size of the LOB must be smaller than, or equal to, the size of the target column. You can set Replicat to truncate the LOB and post it without error by using the DBOPTIONS parameter with the ALLOWLOBDATATRUNCATE option in the Replicat parameter file.
- Are you replicating empty source LOB columns to a Sybase target? Replicat converts empty LOB columns to NULL, but Sybase does not permit null LOB columns. Use the DBOPTIONS parameter with the EMPTYLOBSTRING option to substitute a string value in place of NULL so that the operation is accepted.

✔ Did Replicat return an "incompatible record" error (error 509)?

- Does the error read like this?

```
"2006-09-27 15:45:22 GGS ERROR 509 GoldenGate Delivery for Oracle,
ORAREP.prm: Extract read, Incompatible record in
C:\GOLDENGATE802\DIRDAT\JD000000, rba 5287 (getting header)."
```

- Either the data is corrupted or written in a format other than the GoldenGate internal format. The following are the recommended steps to take.

First step

Did you recently run an initial load? See if FORMATASCII was used for Extract during the load. Other Extract parameters to look for are FORMATSQL and FORMATXML. All of these parameters write data in a format that is incompatible with Replicat.

Next step

Is Extract configured to write over a trail that was used by an old process? Or, are two Extract processes configured to write to the same trail? Possible causes of these problems are:

- ❍ A previously active group was dropped and recreated to link to the same trail name as before. This causes the new Extract to start writing to the beginning of the first file in the trail again, overwriting data that was written by the old group.
- ❍ The parameter file for one Extract group was copied as the basis for the second group, but trail names (and possibly other names) were not changed for the second group. One group is writing over the other's data in the same file.

What happens to cause the Replicat error is this: A complete record is followed by an incomplete record that was overlayed by a different write. Instead of the incomplete record starting with the beginning of a header, as Replicat expects, it starts somewhere in the middle.

To use Logdump to analyze an error 509 condition

1. Issue OPEN <path name> to open the sequence number shown in the error, for example OPEN C:\GOLDENGATE802\DIRDAT\JD000000.
2. Referring to the RBA in the error, use the POS command to look at the timestamps of records before and after that position. If later records have earlier timestamps, those records overwrote existing ones in the trail.

   The best solution is to start over, because data was probably lost or corrupted.

To start replication over again

1. Move the old trail files out to a temporary location, in case you need them later, or in case a GoldenGate support engineer needs to see them.
2. Drop and re-create the Extract groups.
3. Create a new trail for each Extract group, and give each a unique name.
4. Create new Replicat groups.
5. Resynchronize the data. If the source data must remain active while you are resynchronizing the data, refer to the instructions for performing an initial load in the *GoldenGate for Windows and UNIX Administrator Guide*.

   NOTE When a trail is corrupted this way, a debug509.txt file is created in the GoldenGate installation directory. Contact GoldenGate Technical Support and send them the file, or see GoldenGate Knowledge Base public solution 4897 for help with resolving this yourself.

Next step

Did a data-pump Extract abend?

If a data pump abends before it can update its checkpoint file, it processes data from the previous checkpoint when it starts again. This causes it to write the same records over again in the trail. However, those records probably will not match the previous records exactly, and the file will get corrupted.

If this is the case, Replicat is probably stuck in the middle of an incomplete or corrupted record.

To reposition Replicat in the trail

1. Using Logdump's POS and SCANFORHEADER commands, try to determine the time and RBA when Extract started to overwrite the data. Usually a time gap will reveal this position.
2. If you cannot find the gap, position Replicat to the RBA of a record that has a commit timestamp of a few minutes before the timestamp of the record nearest to the RBA where Replicat abended. To do this, issue the following command in GGSCI.

   ```
   ALTER REPLICAT <group>, EXTSEQNO <number>, EXTRBA <RBA>
   ```

3. Enable HANDLECOLLISIONS by issuing the following command.

   ```
   SEND REPLICAT <group>, HANDLECOLLISIONS
   ```

4. Run Replicat until it is past the RBA where it abended. Use the INFO REPLICAT command to view the current RBA.

   ```
   INFO REPLICAT <group>
   ```

5. Disable HANDLECOLLISIONS with the following command.

   ```
   SEND REPLICAT <group>, NOHANDLECOLLISIONS
   ```

   NOTE This process does not ensure data integrity. If the target tables had triggers that inserted additional rows to another table, it may result in extra data.

To skip a record in a trail

If there is a record in the trail that you want to skip, use the following procedure.

1. Open the Replicat parameter file by issuing the following command in GGSCI.

   ```
   EDIT PARAMS <group>
   ```

2. In the file, add the RESTARTCOLLISIONS parameter. This parameter handles duplicate or missing operations.
3. In the file, add the GROUPTRANSOPS or MAXTRANSOPS parameter with a value of 1. This sets Replicat to commit after one operation.
4. Save and close the parameter file.
5. Stop and start Replicat and let it fail on that transaction.

   ```
   STOP REPLICAT <group>
   START REPLICAT <group>
   ```

6. Change directories to the GoldenGate directory, and run Logdump.
7. Open the trail file that contains the record.

   ```
   OPEN <file>
   ```

8. Enable header and detail display.

   ```
   GHDR ON
   DETAIL ON
   ```

9. Use the Logdump SCANFORHEADER command to determine the RBA of the next good record.

   ```
   SCANFORHEADER
   ```

10. In GGSCI, alter Replicat to start at that record.

    ```
    ALTER REPLICAT <group>, EXTRBA <rba>
    ```

11. Edit the Replicat parameter file and comment out, or remove, GROUPTRANSOPS or MAXTRANSOPS.

    ```
    EDIT PARAMS <group>
    ```

12. Save and close the parameter file.

13. Restart Replicat.

    ```
    START REPLICAT <group>
    ```

For more information, see GoldenGate Knowledge Base public solution 4897.

To prevent Replicat from abending on errors

You can specify error-handling rules with the REPERROR parameter. You can use one REPERROR statement to handle most errors in a default way, while using one or more other REPERROR statements to handle specific errors differently. For example, Replicat can ignore duplicate-record errors but abend in other cases. REPERROR can be used as a global parameter or within a MAP statement. For more information, see the *GoldenGate for Windows and UNIX Reference Guide*.

Cannot delete a Replicat group

✔ Are you trying to delete a Replicat group but the DELETE REPLICAT command fails?

- Does the error state "checkpoint table does not exist"?
- Either the group was created without the NODBCHECKPOINT argument, or the checkpoint table was deleted.
- In either case, use the ! argument with DELETE REPLICAT. The ! tells GoldenGate to ignore the checkpoint table. GoldenGate will generate a warning that the database checkpoint could not be deleted, but the group will be removed successfully. The syntax is:

  ```
  DELETE REPLICAT <group> !
  ```

Troubleshooting data-synchronization problems

By default, GoldenGate does not verify that the before images of source and target data are identical before it applies a change to the target. GoldenGate qualifies target rows only by matching the source and target keys, unique-key columns, or substitute key columns defined with the KEYCOLS option of TABLE and MAP. You should test for out-of-sync data on a regular basis.

NOTE Before-image validation can be done by using an external function such as a conflict resolution routine.

Testing for out-of-sync rows

To compare source and target data, use GoldenGate Veridata. You can use GoldenGate Veridata on active source tables while replication (native or third-party) is active. For more information about GoldenGate Veridata, go to the GoldenGate website at http://www.goldengate.com.

Causes of out-of-sync data

Some causes beyond your control:

- user input errors
- application bugs
- system failures
- disk corruption

Some factors within your control:

- DML on target tables (other than replication)
- tables without keys
- character sets
- differences between source and target structure or primary keys

Discrepancies caused by DML on the target

✓ Are transactional applications running on the target?

- Changes to target tables, except by Replicat, cause data discrepancies or errors.

✓ Are there triggers on target tables?

- Triggered operations are replicated from the source. If target triggers reproduce those same operations, they cause errors.
- Disable triggers, alter them to ignore the Replicat user, or use SQLEXEC to disable them on demand by issuing the appropriate database command.

NOTE Triggers affecting tables outside the GoldenGate replication environment are permissible.

Discrepancies caused by tables without keys

✓ Are there target tables that lack a primary key or unique constraints?

- Without a primary key or unique constraints, GoldenGate uses all of the columns as a key. Multiple rows could be updated if they contain identical data.
- Use the KEYCOLS option of TABLE and MAP, or use the DBOPTIONS parameter with the LIMITROWS option to prevent multiple rows from being updated (supported databases only).

Discrepancies caused by character sets

✓ Are source and target character sets different?

- If so, extended ASCII will not synchronize correctly.

- (Oracle) Check the NLS_LANG environment variable on the source and target databases. The source NLS_LANG must be set to the character set of the target, and the target character set must be a superset of the source.

Discrepancies caused by non-matching structures or keys

✔ Was the structure of the source table changed?

- If a source structure changes, including changes in a key definition, you must run DEFGEN to create a new source-definitions file. Otherwise, the same structure changes must be applied to the target.
- If source and target key columns are different, use the KEYCOLS option of TABLE and MAP to define other columns as keys.

Resynchronizing an out-of-sync table

In this procedure, there are two Replicat groups:

- The original group, known as groupA
- A new group that you will create, known as groupB

To resynchronize a table

1. Comment out the out-of-sync table in the original Replicat parameter file.

```
EDIT PARAMS <groupA>
```

2. Stop the original Replicat and start it again so it continues processing the tables that are not out-of-sync.

```
STOP REPLICAT <groupA>
START REPLICAT <groupA>
```

3. Write down the timestamp on the source system.
4. Try to resolve any long-running transactions on the source out-of-sync table. You will be copying the source data to the target, and if transactions are open while the copy is being made, they will not be captured.
5. Start a copy of the source data.
6. Import the copy to the target table.
7. Create a new Replicat group for the out-of-sync table, using BEGIN to start at the source timestamp that you recorded step 3, and using the existing trail for EXTTRAIL.

```
ADD REPLICAT <groupB>, EXTTRAIL <existing trail>,
BEGIN <yyyy-mm-dd hh:mm:ss>
```

8. Create a parameter file for the new group and include HANDLECOLLISIONS.

```
EDIT PARAMS <groupB>
```

9. Start the new Replicat group.

   ```
   START REPLICAT <groupB>
   ```

10. View the lag of the new group until it shows "`At EOF, no more records to process.`"

    ```
    SEND REPLICAT <groupB>, GETLAG
    ```

11. Stop the new Replicat group.

    ```
    STOP REPLICAT <groupB>
    ```

12. Edit the parameter file of the new group to comment out or remove HANDLECOLLISIONS.

    ```
    EDIT PARAMS <groupB>
    ```

13. Start the new Replicat group.

    ```
    START REPLICAT <groupB>
    ```

    The next steps merge the table back with the others so that only one Replicat group is needed again.

14. Stop Extract.

    ```
    STOP EXTRACT <extract_group>
    ```

15. View the lag of each Replicat group until it shows "`At EOF, no more records to process.`"

    ```
    SEND REPLICAT <groupA>, GETLAG
    SEND REPLICAT <groupB>, GETLAG
    ```

16. Stop both Replicat groups.

    ```
    STOP REPLICAT <groupA>
    STOP REPLICAT <groupB>
    ```

17. Uncomment the newly synchronized table in the original Replicat parameter file.

    ```
    EDIT PARAMS <groupA>
    ```

18. Start Extract.

    ```
    START EXTRACT <extract_group>
    ```

19. Start the original Replicat.

    ```
    START REPLICAT <groupA>
    ```

20. Delete the new Replicat that you created.

    ```
    DELETE REPLICAT <groupB>
    ```

Troubleshooting data-mapping

Because table and column mapping can be complex, it is a common cause of processing errors or unexpected results.

- The process report shows where a data-mapping error occurred.
- Issue VIEW REPORT <group> or use GoldenGate Director to view the report.

Table and column names

- ✓ Do the source and target tables exist in the databases?
- ✓ Are they specified correctly in the TABLE or MAP statement?
    - (Sybase and SQL Server) Check spelling and case.
    - (Oracle) Make sure names are fully qualified (schema.table).
- ✓ Do table and column names contain only supported characters?
- ✓ Are any column names within quotes? (They will be interpreted as literals).
    - See the TABLE and MAP documentation in the *GoldenGate for Windows and UNIX Reference Guide* for more details about requirements for table and column names.

Definitions

- ✓ Was a source definitions file created and placed on the target system?
- ✓ Is the file referenced correctly with the Replicat parameter SOURCEDEFS?
- ✓ Are source columns mapped to target columns of a similar data type?
    - If types differ, use a column-conversion function in the mapping statement to prevent "mismatched data type" errors.
- ✓ Are source and target columns different sizes?
    - GoldenGate truncates or rounds target data if the target column is smaller than the source column.
- ✓ Are comparisons configured realistically?
    - For example, comparing a numeric column to a numeric value.

Column-missing error

- ✓ Are you using KEYCOLS?
    - (Oracle and SQL Server only) Are the specified columns being logged to the transaction log? Enable logging of the columns. For an Oracle table, you can use ADD TRANDATA with the COLS option. For SQL Server, use the ADD TRANDATA command to enable the additional log information that GoldenGate needs to reconstruct update operations.

✔ Are you using KEYCOLS for both the source and target tables?

- Corresponding source and target key columns must match.

✔ Did you include the columns of an existing primary key or unique index in the KEYCOLS definition?

- Those columns must be included in KEYCOLS.

✔ Did the error occur on a filter or function?

- This indicates that data is missing from the transaction record. Try one of the following:
    - Filter on key columns
    - Enable logging of required columns
    - Use FETCHCOLS or FETCHCOLSEXCEPT in the TABLE statement

✔ Are you using valid filter or comparison elements?

- For example, FILTER takes a numeric value, and WHERE takes a string but not arithmetic operators or floating-point numbers.

Fetch failures on Oracle source

✔ Was the source row that contained the data deleted?

✔ (Oracle 9i and later) Has the undo retention expired?

- If so, then the read-consistent image that Extract needs is gone, and a snapshot-too-old message will be returned.
- Add the FETCHOPTIONS parameter with the NOUSESNAPSHOT option to Extract. This parameter tells Extract to fetch from the table, not the undo segments. Once GoldenGate is past the affected record, remove the parameter so that Extract resumes fetching from the undo segments.
- If possible, increase the amount of time that Oracle keeps information in the undo segments.

NOTE Certain DDL operations that alter the structure of a table, such as a table truncate, can cause fetch errors too.

✔ Are you using NOUSELATESTVERSION for FETCHOPTIONS?

- If so, GoldenGate cannot default to fetching from the table if the undo record is not available. Unless error handling options are set with that parameter, Extract abends.
- If fetched data is available but cannot be merged with the redo data, then Extract abends.

Replicat errors on Sybase targets

✔ Are you attempting to map TIMESTAMP columns?

- TIMESTAMP columns must be excluded from GoldenGate extraction. These data types are populated automatically by the database, which generates errors if Replicat attempts to post a timestamp value that was replicated from the source record. To exclude timestamp columns from being captured by GoldenGate, use the COLSEXCEPT option of the TABLE parameter. Note: Because the system generates these data types, the source and target values will be different.

Escaped characters not converting correctly

✔ Are you using an octal or hexadecimal escape sequence to represent native encoding or Unicode characters in a parameter file?

- An octal sequence must contain exactly three digits. A hexadecimal sequence must contain exactly two digits. Any more than the expected number of digits causes the sequence to be interpreted as multiple characters.
- For more information about using escape sequences, see TABLE or MAP in the *GoldenGate for Windows and UNIX Reference Guide*.

Troubleshooting SQLEXEC

✔ Does the Extract or Replicat user have permissions to call and execute database procedures and commands?

✔ Are you trying to use input or output parameters for a SQLEXEC statement at the root of a parameter file?

- Parameters can only be used in SQLEXEC statements within a TABLE or MAP statement.

✔ Does parameter resolution result in a column-missing error?

- (Oracle and SQL Server) Is the referenced column being logged to the transaction log? Enable logging with the ADD TRANDATA command. For an Oracle table, you also can use ADD TRANDATA with the COLS option.
- Is the stored procedure supposed to execute more than once, for example to specify the same stored procedure for two different column maps? In that case, use the ID <logical name> option of SQLEXEC to specify a logical name for each execution, similar to that shown in bold face here:

```
TABLE sales.srctab, TARGET sales.targtab,
SQLEXEC (SPNAME lookup, ID lookup1, PARAMS (param1 = srccol)),
COLMAP (targcol1 = lookup1.param2),
SQLEXEC (SPNAME lookup, ID lookup2, PARAMS (param1 = srccol)),
COLMAP, (targcol2 = lookup2.param2);
```

- Is the stored procedure supposed to synchronize a source row with multiple target rows? Set the EXEC option of SQLEXEC to SOURCEROW so that it executes for each source-row operation.
- To prevent a process from abending on SQLEXEC errors, specify error handling with the ERROR option.

To trace SQLEXEC problems

Use the TRACE option of SQLEXEC. This option has two modes:

- ALL writes all input and output parameters to the discard file. This is the default.
- ERROR writes the input and output parameters to the discard file only after a SQL error occurs.

The discard file shows the parameters that are being passed, and the report file shows the number of attempts that were made to pass them.

Sample parameter file that includes tracing:

```
MAP MASTER.INTERACTION_ATTR_VALUES , &
TARGET MASTER.INTERACTION_ATTR_VALUES, &
SQLEXEC ( SPNAME ggs_conflict_rule, ID ggs_interaction_attr_values, &
   PARAMS ( LMS_TABLE = "INTERACTION_ATTR_VALUES", &
   KEY1 = interaction_attribute_value_id, &
   KEY2 = attribute_value_seq), &
   ERROR RAISE ,TRACE ALL ) ;
```

Sample discard file with SQLEXEC tracing enabled:

```
Input parameter values...

LMS_TABLE: INTERACTION_ATTR_VALUES
  KEY1: 2818249
  KEY2: 1
Report File:

From Table MASTER.INTERACTION_ATTR_VALUES to
MASTER.INTERACTION_ATTR_VALUES:
       #  inserts:      0
       #  updates:      0
       #  deletes:      0
       #  discards:     1

  Stored procedure GGS_INTERACTION_ATTR_VALUES:
       attempts:        2
       successful:      0
```

Troubleshooting missed transactions

✔ Are the affected tables properly specified?

- Verify that the tables are included in TABLE and MAP statements.
- Check syntax: If fully qualified names (owner.table) are used for TABLE, then fully qualified names must be used for MAP. The name in TABLE, an Extract parameter, is written to the trail record header, and Replicat expects an exact match in the source

portion of the Replicat MAP statement. Otherwise, nothing is replicated. Qualifying a table name by including the owner is recommended for GoldenGate versions 8.0 and later, and is required for Oracle tables regardless of the GoldenGate version.

✔ Did you recently run an initial load and use the same parameter file for your load and online groups?

- Check the parameters. After a load, some parameters from the load configuration must be altered to suit the online configuration. A common mistake: Using RMTFILE for the load, but forgetting to change it to RMTTRAIL for the online group.

To resolve the problem

1. Stop Extract.
2. Edit the parameter file to fix or add parameters.
3. Add the remote trail with ADD RMTTRAIL in GGSCI, if you have not done so already.
4. Start Extract.

✔ Are LOBs not being captured?

- (Oracle 8i) For LOBs to be captured in Oracle 8i, other columns in the row must be modified in the same transaction.
- Solution: Write a stored procedure to set a column value to itself.

✔ Is the database Sybase, and is the secondary truncation point disabled?

- GoldenGate manages the secondary truncation point. If Extract is stopped for an extended period of time, the secondary truncation point does not move, which prevents transaction data from being purged. Ultimately, the log fills up, and the database halts.
- Solution: The only thing that you can do if the log fills up is to disable the secondary truncation point and finish purging the log. The purge will remove entries that were not yet read by Extract, so you must resynchronize the source and target data. Afterward, initialize the secondary truncation point by issuing the following database command before starting Extract.

```
dbcc settrunc( 'ltm', ‘valid’)
```

✔ Was transaction data configured for replication?

- (Applies to most supported databases) Was ADD TRANDATA issued for each table? This GGSCI command makes modifications to database settings so that GoldenGate can obtain the information it needs from the transaction logs.

An example of an Oracle error reflecting lack of ADD TRANDATA is:

```
Missing one or more key fields in update
Missing one or more key fields in update
Aborting transaction beginning at seqno 0 rba 10008
                         error at seqno 0 rba 10008
Problem replicating GGS.TCUSTMER to GGS.TCUSTMER_T
Mapping problem with compressed update record (target format)...
*
CUST_CODE = NULL
NAME = mike
```

For more information about ADD TRANDATA, see the GoldenGate installation guide for your database and the *GoldenGate for Windows and UNIX Reference Guide*.

To include missed transactions in the target

To update the target with missed operations after running ADD TRANDATA or setting data capture changes, you need to resynchronize the affected tables.

To skip missed transactions

To skip the transactions that were missed before running ADD TRANDATA, follow this procedure. The target will not reflect accurate data in this case.

1. Stop Replicat.

   ```
   STOP REPLICAT <group>
   ```

2. Close the current trail file and open a new one.

   ```
   SEND EXTRACT <group> ROLLOVER
   ```

3. Point Replicat to the new file.

   ```
   ALTER REPLICAT <GROUP NAME>, EXTSEQNO <n>, EXTRBA 0
   ```

   Where: <n> is the sequence number of the new file.

4. Start Replicat to start processing from the new file.

   ```
   START REPLICAT <group>
   ```

Troubleshooting problems with Oracle DDL replication

Extract abends with “OCI Error executing fetch ...”

✔ Does the error report an invalid identifier, similar to the following?

```
OCI Error executing fetch for <table_name>: 904-ORA-00904: <column name>:
invalid identifier …
```

- This error happens if key columns are changed before GoldenGate can process updates or deletes that were issued on the old column structure. In this context, the *key columns* are whichever columns are being used by GoldenGate as the key. This could be a primary key, a unique key or index, a substitute key defined with a KEYCOLS clause, or all of the columns in the absence of those other identifiers.

No DDL is captured by Extract

✔ Is there a valid DDL parameter in the Extract parameter file?

- If so, continue with the next diagnostic step.

✔ Is the DDL trigger enabled?

- Does the DDL trace file (default name of ggs_ddl_trace.log) show the following message?

```
DDL operation failed because RECYCLEBIN is turned on. Recycle bin must
be turned off by setting in parameter file: RECYCLEBIN=OFF (10gr2 and up)
or _RECYCLEBIN=FALSE (10gr1), and restarting database.
```

  This indicates that the Oracle recycle bin is on, so the DDL trigger failed. The recycle bin must be disabled for GoldenGate DDL processing. Follow the instructions in the error message and then enable the trigger by running the ddl_enable script from the GoldenGate directory.

✔ Does the value of the GGSCHEMA parameter in the GLOBALS file match the name of the schema that was used to install the GoldenGate DDL objects?

✔ Are you still unable to determine why DDL is not being captured? Use the following procedure to verify that the DDL objects are installed and are operating correctly.

To diagnose failure to capture DDL

1. In the root GoldenGate directory, open the GLOBALS file.
2. Make a note of the GoldenGate DDL user name and, if applicable, any non-default names specified with the DDLTABLE and MARKERTABLE parameters.
3. Connect to the source database as SYSDBA.
4. Run the marker_status.sql script. This script verifies that the GoldenGate marker table is installed. It should show the following results.

```
Please enter the name of a schema for the GoldenGate database objects:
ggsuser
Setting schema name to GGSUSER

MARKER TABLE
-------------------------------
OK

MARKER SEQUENCE
-------------------------------
OK
```

5. Run the ddl_status.sql script. This script verifies that the GoldenGate DDL objects are installed.
6. In the script output, look for the location of the DDL trace file, as shown in the following example, and make a note of it.

```
LOCATION OF DDL TRACE FILE
--------------------------
C:\ORACLE\ADMIN\TEST10G1\UDUMP/ggs_ddl_trace.log
```

7. At the end of the output, verify that the following shows successful installation of the DDL objects.

```
SUCCESSFUL installation of DDL Replication software components
```

8. Run GGSCI.
9. In GGSCI, stop Extract.

```
STOP EXTRACT <group>
```

10. Edit the ddl_tracelevel.sql script that resides in the root GoldenGate directory, and set the trace level to 1.
11. Start Extract again.

```
START EXTRACT <group>
```

12. On the source, issue a DDL statement as a test.
13. Open the trace file in a text editor.
14. At the bottom of the trace file, search for a line that contains the date, time and DDL text, for example:

```
... 08-10 11:08:16 : DDL : DDL operation [create table qatest1.testTable
(<n> Number) ], sequence [17047] ...
```

    If this line exists, it means that the DDL trigger captured the DDL.

15. Make a note of the sequence number.
16. Proceed to the next step to confirm that the record exists in the DDL history table.
17. Connect to the database as SYS.
18. Issue the following query in SQL*Plus, using the sequence number you recorded in place of the one shown here.

```
select metadata_text from <ggsuser>.<ggs_ddl_hist> where seqno=17047;
```

    Where:

    - <ggsuser> is the GoldenGate DDL user name that is listed in the GLOBALS file that resides in the GoldenGate root directory.
    - <ggs_ddl_hist> is either the name listed with DDLTABLE in the GLOBALS file or else the default DDL history table name of ggs_ddl_hist.

19. In the output, look for a record that contains the DDL statement that you issued.

20. If the record exists, keep the query output open, and then proceed to the next step to determine the marker sequence number.

21. Starting from the line you read in the previous step, search for the first occurrence of MD_TAB_MARKERSEQNO. You should see something like the following, which shows the marker sequence number.

    ```
    ... MD_TAB_MARKERSEQNO(key1 = [] key2 = []) = [16549] ....
    ```

22. If the line exists, do the following:
    - Record the marker sequence number.
    - Copy the query statement and paste it into a text file for future use.
    - Proceed to the next step to verify that the marker record exists in the marker table.

23. In SQL*Plus, issue the following query, using the sequence number you recorded in the previous step.

    ```
    select metadata_text from <ggsuser>.<ggs_marker> where seqno=16549;
    ```

    Where:

    - <ggsuser> is the GoldenGate DDL user name that is listed in the GLOBALS file that resides in the GoldenGate root directory.
    - <ggs_marker> is either the name listed with MARKERTABLE in the GLOBALS file or the default DDL marker table name of ggs_marker.

24. In the output, look for the DDL statement that you issued. If the statement is there, a marker was written to the redo logs as expected. Do the following:
    - Copy the query statement and paste it into the text file with the other query.
    - Proceed to the next step to verify that Extract can read the marker.

25. In SQL*Plus connect to the database as the GoldenGate DDL user, the one listed in the GLOBALS file.

26. Issue the following queries, which are the two that you copied and pasted to the text file.

    ```
    select metadata_text from <ggsuser>.<ggs_ddl_hist> where seqno=17047;
    select metadata_text from <ggsuser>.<ggs_marker> where seqno=16549;
    ```

27. Verify that these queries returned the same results as the ones you issued as the SYS user. Do one of the following:
    - If you do not get output, but instead get a database error, this indicates that the ddl_rolesetup.sql script was not executed properly to grant the GoldenGate DDL user the correct privileges. Run the role_setup script as SYSDBA and then grant the role that you just created to all GoldenGate users under which the following GoldenGate processes run: Extract, Replicat, GGSCI, and Manager.
    - If you receive valid results from the query, proceed to the next step.

28. In GGSCI, start Extract. Start the existing group, not a new or recreated one. Otherwise, the test DDL statement will not be captured.

    ```
    START EXTRACT <group>
    ```

29. Look for a statement similar to the following.

```
TABLE resolved (entry GGSUSER.GGS_MARKER):
  TABLE GGSUSER.GGS_MARKER;
```

30. If you do not see this statement, or if you get an error, contact GoldenGate Technical Support. Or, reinstall the DDL environment, understanding that you will lose any DDL operations that have not been captured.

Specific DDL is not captured or excluded

✔ Are ALTER DATABASE and ALTER SYSTEM being excluded?

- This is normal. These operations are not considered to be DDL by the database and are ignored.

✔ Are other specific operations that are supported by GoldenGate not being captured?

- Check the structure of the DDL parameter statement.
  - First, make certain that the operation is listed in an INCLUDE OPTYPE statement or else is *not* listed in an EXCLUDE OPTYPE statement, and verify that the missing operation does not apply to an object whose name was in an EXCLUDE OBJNAME statement.
  - Perhaps you have the same operation type or object name in both an EXCLUDE and INCLUDE statement. If so, the EXCLUDE takes priority over the INCLUDEs, and this might be why some operations are not being captured or applied to the target.
  - Or, perhaps you are only replicating objects that are MAPPED in scope, and the missing DDL was for objects that are UNMAPPED in scope.

✔ Are you using a DDL EXCLUDE INSTR or a DDL INCLUDE INSTR?

- The filter criteria may not be inclusive or exclusive enough. For example, if you filter on INSTR 'supplemental log', it will filter both ADD SUPPLEMENTAL LOG and DROP SUPPLEMENTAL LOG statements.
- Check for spaces in the source DDL. If the source includes spaces, then you must include those spaces in your filter specification.

Data on new, altered, and renamed tables not captured

✔ Did you forget to use the DDLOPTIONS ADDTRANDATA parameter statement or issue the ADD TRANDATA command in GGSCI for the new source table name?

- The ADD TRANDATA functionality of GoldenGate creates a supplemental log group for a new table that is to be part of the GoldenGate configuration. The supplemental logging provides information that GoldenGate needs to capture transactional data from the logs.
- Unless you either enable supplemental logging for a new table through the database interface, or else use one of the ADD TRANDATA options described here, the data is not captured by the Extract process.

- When DDL support is enabled, it is recommended that you use the DDLOPTIONS ADDTRANDATA statement, instead of the ADD TRANDATA command in GGSCI. By using the parameter, you can ensure that supplemental logging is created or updated automatically for all CREATE, ALTER, and RENAME operations that affect tables in the DDL configuration. It also deletes the old supplemental log groups when a table is renamed, preventing the need for a manual procedure.

✔ Did you use a DDLOPTIONS ADDTRANDATA statement and data still is not captured?

- There might have been a lag between when the table was created or altered, and when the ADD TRANDATA took effect. If there was DML issued during this interval, it was not captured because the supplemental logging was not yet added or updated.

✔ Are you using TABLEEXCLUDE with the NORENAME option or DDLOPTIONS with the NOCROSSRENAME option?

- If so, were the table names previously of UNMAPPED scope, but the new names are of MAPPED scope? You might see one of the following messages in the Extract report:

```
Object [FIN.ACCT] was renamed even though it's marked excluded with
NORENAME….
```

or

```
Object that is not replicated [FIN.ACCT] was renamed into object that is
[FIN.SALES].
```

- Those renames will succeed, but data in those tables might not be replicated to the target.

RENAMEs converted to ALTER TABLE RENAME

✔ Is the message similar to the one below?

```
DDL RENAME found, old object [JCAN.OLDT], new object [JCAN.NEWT], RENAME
converted to ALTER TABLE, new operation [ALTER TABLE JCAN.OLDT RENAME TO
NEWT]
```

- This is normal. GoldenGate converts RENAME TABLE to ALTER TABLE RENAME to work around permissions problems that can occur if the source and target GoldenGate database user names are different from each other.

RENAME operations are failing

✔ Are the old or new table names longer than 16 characters?

- Oracle only allows 16 characters plus the double quote marks (18 total) for RENAME and ALTER TABLE RENAME, because of the ANSI limit for identifiers.

Comments are in the wrong place

✔ Were the comments in the middle of the name of the source object in the source DDL statement?

- If so, it is expected behavior for GoldenGate to move them to the end of the object name on the target. This does not affect the integrity of the DDL replication.

Derived object names are not being converted correctly

✔ Is the DDLOPTIONS parameter being used with the NOMAPDERIVED option?

- If so, then regardless of whether you explicitly or implicitly mapped the derived object, NOMAPDERIVED overrides the mapping and prevents the conversion.

✔ Is the operation a rename?

- Check your mappings. For RENAME and ALTER TABLE RENAME, the new table name is considered the base object name, and the old name is the derived object name.

String substitution not working

✔ Does your DDL INCLUDE statement include the object for which you want to use string substitution? Or, is that object excluded somehow in an EXCLUDE statement?

- The DDL statement always executes first, regardless of the ordering of the DDL and DDLSUBST parameters. Did you perhaps create a DDLSUBST statement that affects a non-included object, thinking it would execute independently of the INCLUDE or EXCLUDE in the DDL parameter?

✔ Is there a DDLOPTIONS REMOVECOMMENTS BEFORE in use?

- This parameter removes comments before the point in processing where string substitution occurs. Use DDLOPTIONS REMOVECOMMENTS AFTER if you want comments removed on the target after string substitution occurs. To prevent the removal of comments, do not use a REMOVECOMMENTS option.

✔ Are there spaces in the source DDL statement?

- The DDLSUBST specification should also include those spaces.

DDL processing steps not in the process report

✔ Is reporting enabled?

- By default, DDL runtime information is not included in the report because it expands the size of the file.
- To enable DDL runtime reporting, use a DDLOPTIONS REPORT statement in the parameter file of the process for which you want the reporting to be generated (Extract, Replicat, or both).

Replicat abends on ggs_ table

✔ Is the error similar to the following?

```
GGS ERROR 118 Target table not resolved for source [GGS.GGS_SETUP].
```

- This indicates that one or more GoldenGate DDL objects are included in the replication configuration. These objects are:
  - GGS_MARKER
  - GGS_DDL_HIST
  - GGS_SETUP
  - GGS_DDL_OBJECTS
  - GGS_DDL_COLUMNS
  - GGS_DDL_LOG_GROUPS
  - GGS_DDL_PARTITIONS
  - GGS_DDL_PRIMARY_KEYS
- Do not include the GoldenGate DDL objects in any DDL, TABLE or MAP parameter statements, and make certain that wildcard specifications in those parameters do not include them. These objects must be excluded from the GoldenGate replication configuration.

The database returns ORA-04021

✔ Is the error text like the following?

```
"timeout occurred while waiting to lock object SYS.DATABASE"
```

- This error can occur if any changes were made to the GoldenGate DDL objects while sessions that performed DDL operations were still open.
- Close all sessions in the database that performed DDL. If that does not help, close all database sessions. Then reinstall the GoldenGate DDL trigger, using the NORMAL option.
- If this procedure does not help, you need to reinstall the DDL objects.
- Refer to the *GoldenGate for Windows and UNIX Administrator Guide* for procedures.

The DDL objects tablespace filled up

✔ Is the error text like the following?

```
ERROR at line 1:
ORA-00604: error occurred at recursive SQL level 1
ORA-01653: unable to extend table JSUN.GGS_MARKER by 128 in tablespace USERS
ORA-06512: at line 647
ORA-01653: unable to extend table JSUN.GGS_MARKER by 128 in tablespace USERS
```

- The tablespace that contains the GoldenGate DDL objects is full. When the tablespace is full, users cannot perform DDL across the whole database system, even for DDL that is executed in other tablespaces.

To resolve this problem

1. Stop all DDL.
2. Disable the GoldenGate DDL trigger.
3. Add more storage to the tablespace.
4. Enable the DDL trigger.
5. Start the GoldenGate process.

For more information about how to enable and disable the DDL trigger, see the DDL chapter in the *GoldenGate for Windows and UNIX Administrator Guide*. This chapter also contains information on how to purge the history and marker tables to control their size.

Troubleshooting file-maintenance problems

Process report not generated

✔ Did a GoldenGate process abend without generating a report?

- This happens occasionally. See “Using the process report” on page 13 for more information and a resolution.

Discard file not created

✔ Did you specify a location for the file with the DISCARDFILE parameter?

- GoldenGate does not create a discard file by default.

✔ Are you looking in the correct directory?

- See the DISCARDFILE parameter for the location.

✔ Could there be a security violation?

- Perhaps the user that wrote the last discard file has a security level that won't let the new one be created.

Discard file too big

✔ First, try to resolve whatever problem is causing discarded records to accumulate in the file. The records themselves can provide information that helps you diagnose it.

✔ If you can tolerate the errors, you can control the size of the file with the following parameters:

- DISCARDROLLOVER: specifies parameters for aging the file
- MAXDISCARDRECS: limits the number of errors written to the file

- DISCARDFILE with the PURGE option: purges the file before writing new content
- DISCARDFILE with the MEGABYTES option: sets a different maximum file size (default is 1 MB).

Trail file is not emptying

✔ Is it a local trail?

- If a local trail is not emptying, check for TCP/IP issues that delay transfer of data across the network.

✔ Is it a remote trail?

- If a remote trail is not emptying, find out if Replicat is running and processing data.
- If yes, Replicat might be processing a large transaction.
- You can see which tables have large transactions by using the TRANSHIST command in Logdump. See Chapter 4.
- Consider partitioning tables that have large transactions into multiple trails and Extract or Replicat groups, or use data pumps. Use the FILTER option in the TABLE or MAP statements, and use the @RANGE function to divide the rows among the different groups. For more information about using multiple groups and row ranges, see page 22.

Trail not rolling over

✔ What is the maximum file size for the trail?

- To check the maximum file size, use the INFO EXTTRAIL * or INFO RMTTRAIL * command and view the File Size field. Maybe the file size has not reached that value yet.
- To change the file size, use ALTER EXTTRAIL or ALTER RMTTRAIL with the MEGABYTES option.

Trail files not being purged

✔ Are you using PURGEOLDEXTRACTS to manage the trail?

- If not, add PURGEOLDEXTRACTS to the Manager parameter file to prevent old files from accumulating.
- If you are using PURGEOLDEXTRACTS, make certain that the Manager user has the authority to purge trail files, and make certain that the PURGEOLDEXTRACTS options are used correctly. See the *GoldenGate for Windows and UNIX Reference Guide.*

✔ Is there an obsolete Replicat group that is linked to the trail?

- A trail file will not be purged if another process has a checkpoint in it. Delete the obsolete group with the DELETE REPLICAT command, so that the checkpoint records are deleted.

- If a checkpoint table is being used for the group, log into the database with the DBLOGIN command first, so that the checkpoint will be removed from the table.

```
DBLOGIN [TARGETDB <dsn>,] [USERID <user>, PASSWORD <pw>]
DELETE REPLICAT <group>
```

NOTE Note: In the DBLOGIN command, TARGETDB or USERID might be required, or both will, depending on the database type and what kind of authentication is required.

Trail files purge too soon

✔ Are multiple Replicat groups linked to the same trail? Or are a Replicat and a data pump linked to the same trail?

- Are you using PURGEOLDEXTRACTS?
- If so, how are you using it? When more than one process reads a trail, PURGEOLDEXTRACTS should only be used as a Manager parameter, not as an Extract or Replicat parameter. When you use PURGEOLDEXTRACTS as a Manager parameter, there are better options to defer purging until all processes are done with a file.

Report file too big

✔ Are you using macros?

- If so, are you using CMDTRACE?
- This parameter causes inclusion of macro expansion steps in the report. The DETAIL option writes even more data.
- If your reporting requirements do not include macro tracing, disable CMDTRACE to reduce the report size.

✔ Are you using LIST?

- This parameter lists the macros of a library in the report. If your reporting requirements do not include macro lists, remove LIST to reduce report size. Otherwise, you can control report size by setting an aging schedule with the REPORTROLLOVER parameter.

✔ Are you using DDLOPTIONS REPORT?

- If DDL information is included in a process report, it increases the size of the file. If you do not need to see step-by-step processing history of DDL operations in the report file, remove the DDLOPTIONS REPORT parameter. The default is DDLOPTIONS NOREPORT.

Troubleshooting other problems

Too many open cursors

✔ Is the maximum for open cursors reached?

- The Extract process maintains cursors for queries that fetch data. The Replicat process maintains cursors for caching SQL statements. These processes also might use other cursors, such as those needed for SQLEXEC operations.
- If the database will support additional cursors, you can increase the GoldenGate allocation with the Extract parameter MAXFETCHSTATEMENTS or the Replicat parameter MAXSQLSTATEMENTS. If the database will not support an increase, use those parameters to reduce the number of cursors. This will affect performance because GoldenGate will need to age more statements.

Time differences

✔ Is the lag report inaccurate?

- Lag is always accurate in absolute terms, but may seem inaccurate because of differences in system time on source and target machines.
- Use the TCPSOURCETIMER parameter to reflect a more accurate lag result.

✔ Do timestamps seem inaccurate?

- GoldenGate records are extracted in GMT (Greenwich Mean Time) format. However, source and target timestamps could differ for the same set of data if an application uses the timestamp recorded by GoldenGate as a column in the target record. Discrepancy occurs if:
  - the timestamp is converted to Local Civil Time (LCT) before being included in the database record, and that data is replicated to a location in another time zone.
  - timestamps are derived locally and used in a key, which can create sequence differences if a table contains records created in two different time zones.
- Solution: If using timestamps written by GoldenGate in application data, maintain all timestamps as GMT offsets. GMT is reliable in any time zone and through Daylight Saving time changes without sequence problems. If that is not possible, then use GoldenGate conversion functions to change the format back and forth as needed.

Oracle ADD TRANDATA problems

✔ Are you getting errors with ADD TRANDATA?

- Is the error “ No tables matched specification”?
  - Did you type the table name correctly?
  - Does it exist in the database?
  - Did you use a fully qualified table name? When a schema name is not specified, GoldenGate uses the current user as the schema.

- Is the error “SQL error (-942). ORA-00942: table or view does not exist.”?
  - Was the command issued for an Oracle 8i database? For Oracle 8i tables, ADD TRANDATA creates a trigger on the table. The user running ADD TRANDATA might not have privileges to create the trigger. Grant permission, and the command should work.

Other Oracle-specific problems

✔ Are there excessive operations being processed by an Oracle Extract?

- Did you change storage for a SECUREFILE LOB?
- 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.

✔ Are you replicating sequences with GoldenGate, but the target sequences are not in the expected range of values?

- By default, Replicat checks to see if the target sequences are either higher or lower than those of the source, depending on the direction of the incrementation method that is in use. If there is a discrepancy, Replicat corrects it automatically unless NOCHECKSEQUENCEVALUE is used in the parameter file.
- If NOCHECKSEQUENCEVALUE is not in use, and you cannot find a possible cause outside the GoldenGate configuration, please contact GoldenGate Technical Support.

Sybase missing server messages

✔ Are Sybase server messages missing from the error log?

- By default, certain information messages and messages with a severity level greater than 10 are not logged by GoldenGate to the error log.
- To show these messages in the log, use the Extract parameter DBOPTIONS with the SHOWINFOMESSAGES and SHOWWARNINGS options.

DB2 z/OS-specific problems

✔ Is the number of operations that were executed on the database different from the number that GoldenGate shows were extracted?

- These statistics will vary because of the way that messages are updated in the DB2 logs. DB2 does not log update statements if they do not physically change a row.

✔ Is GoldenGate issuing commit locks?

- This can happen if the DB2 parameter mvsattachtype is set to CAF (call attachment facility, the default). CAF does not support multiple connections, so GoldenGate issues commit locks on the system catalog tablespace until it receives the commit for its open connection. To avoid commit locks, set the attachment type to RRSAF (Resource Recovery Services Attachment Facility), if possible.

✔ Is there an authorization failure 00D31024?

- If the error is similar to the one following, it means that the location name of the local DB2 subsystem needs to be inserted into the DB2 remote server locations table, SYSIBM.LOCATIONS.

```
2006-01-09 15:48:45 GGS WARNING 228 ODBC operation failed: Couldn't
connect to cdnxdb2q.
2006-01-09 15:48:45 GGS WARNING 228 {DB2 FOR OS/390}{ODBC DRIVER}
DSNT408I SQLCODE = -922, ERROR: AUTHORIZATION FAILURE: 00D31024 ERROR.
REASON CONNECT DSNT418I SQLSTATE = 42505 SQLSTATE RETURN CODE DSNT415I
SQLERRP = DSNLVINF SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD = 0 0 0 -1 0 0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD = X'00000000' X'00000000' X'00000000' X'FFFFFFFF'
X'00000000' X'00000000' SQL DIAGNOSTIC INFORMATION ERRLOC=1:13:2.
2006-01-09 15:48:45 GGS ERROR 118 ODBC error 42505, native database error
-922.
2006-01-09 15:48:45 GGS ERROR 190 PROCESS ABENDING.
```

- A statement similar to the following adds the DB2 location name to the remote group:

```
INSERT INTO SYSIBM.LOCATIONS (LOCATION, PORT) VALUES ('DB2A', '446');
```

✔ Is GoldenGate unable to use RRSAF?

- On a Sysplex, RRS is sensitive to system crashes. RRS might start normally after a crash, but GoldenGate might not be able to use RRSAF. You might see messages similar to:

```
ODBC error 58004, native database error -99999***** ABEND******
ABEND****** {DB2 FOR OS/390}{ODBC DRIVER}  SQLSTATE=58004 ERRLOC=2:170:4;
RRS “IDENTIFY” failed using DB2 system:DB7G,  RC=0c and REASON=00f30006
```

- To resolve this error, shut down RRS and then rebuild the RRS structures in the Coupling Facilities using your RRS installation job.

✔ Did Extract fail and return DB2 reason codes 00C90063 and 00C90064?

- This can be a symptom of mismatched compression dictionaries. You will need to resynchronize the affected data and start GoldenGate processing from the initialized state.

✔ Is GoldenGate returning the following message?

```
2009-06-16 07:32:18  GGS WARNING     228  The current ODBC session does not
support multiple active transactions. GoldenGate recommends using multiple
transactions to ensure transaction integrity and to enable releasing catalog
locks as soon as possible. The ODBC initialization file should have both
'MVSATTACHTYPE=RRSAF' and 'MULTICONTEXT=1'.
```

- This message indicates that the Call Attachment Facility (CAF) is being used, instead of the Resource Recovery Services Attachment Facility (RRSAF). This message can mean one of the following:
  - ❍ The ODBC initialization file specified MVSATTACHTYPE=CAF or did not include an MVSATTACHTYPE specification.
  - ❍ The ODBC initialization file does not contain a section for the DB2 subsystem.
  - ❍ The ODBC initialization file is not valid because it the encoding is not IBM-1047.
  - ❍ The ODBC initialization file is not readable (indicates a file permissions problem).
  - ❍ No ODBC initialization file was specified. There are several ways to specify which ODBC initialization file should be used; GoldenGate sets the DSNAOINI environment variable to the path to the ODBC initialization file that it needs to use. For example, "export DSNAOINI=/etc/odbcDB9A.ini".

SQL Server-specific problems

✔ Is Extract generating a warning that says "replication is not enabled?"

- Are you using the MANAGESECONDARYTRUNCATIONPOINT parameter so that SQL Server manages the secondary truncation point?
- If so, and if this is the configuration that you want, then did you also issue the ADD TRANDATA command for the tables? If so, you will get this message, because Extract expects the NOMANAGESECONDARYTRUNCATIONPOINT in that case. The message is generated as part of the truncation point management process.

✔ Is Extract generating an error that states ""Updates are not supported on tables that do not have TRANDATA added?"

- If the source database is SQL Server 2000, ADD TRANDATA sets a flag on the sysobjects table that tells SQL Server to log full before and after images. This flag is shared by update triggers and by replication. If an update trigger on a table in the Extract configuration is dropped, it will generate this error. Dropping the trigger drops the extended logging for that table. To enable capture of the table again, use the following procedure. This procedure assumes you can stop activity on the source table. If you cannot stop activity, see "To resynchronize a table" on page 53 for instructions.

1. Take the table out of the Extract TABLE parameter.
2. In GGSCI, issue the ADD TRANDATA command for the table.
3. Stop all activity on the source table.
4. Restart Extract and let it capture backlogged transaction data until it is current again. The SEND EXTRACT command with the STATUS option should return "At EOF."

5. Stop Extract.
6. Resynchronize the source and target tables.
7. Restart Extract.

Teradata module not found

✔ Is there a "module could not be found" error?

- The error appears as follows only for Teradata on a Windows machine running Windows XP:

  ```
  2006-08-22 16:48:15  GGS ERROR    520  VAMRDR-520 DLL LoadLibrary [Error
  126:The specified module could not be found.:occurred loading
  C:\GGS\TERA\TAM.DLL].
  2006-08-22 16:48:15  GGS ERROR    190  PROCESS ABENDING.
  ```

- Solution: Put the MSVCR71D.dll driver in your C:\WINDOWS\SYSTEM32 folder.

CHAPTER 4

Using the Logdump Utility

Logdump is a GoldenGate utility that enables you to search for, filter, view, and save data that is stored in a GoldenGate trail or extract file. To avoid any adverse effects on the data or checkpoints in your trails, use Logdump only with guidance from a GoldenGate support analyst.

Getting started with Logdump

This section introduces you to basic Logdump commands that enable you to open files, control the display, navigate through a file, and filter for specific information, among other basic tasks. It also illustrates and explains the components of a record.

Following this section is an alphabetical reference of the Logdump commands.

Viewing the first record

These steps show you how to set up the Logdump environment and start viewing records.

To run Logdump

Run the logdump program from the GoldenGate installation location. Logdump command lines are numbered so that you can use edit and history commands.

To set up the view

The following commands set up a Logdump environment that shows the information most commonly used when analyzing GoldenGate trail records.

1. To view the record header with the data:

   ```
   Logdump 1> GHDR ON
   ```

   The record header contains information about the transaction.

2. To add column information:

   ```
   Logdump 2> DETAIL ON
   ```

   Column information includes the number and length in hex and ASCII.

3. To add hex and ASCII data values to the column information:

   ```
   Logdump 3> DETAIL DATA
   ```

4. To view user tokens:

   ```
   Logdump 4> USERTOKEN ON
   ```

   User tokens are custom user-defined information that is specified in a TABLE mapping statement and stored in the trail file for specific purposes.

5. To control how much record data is displayed:

   ```
   Logdump 5> RECLEN <length>
   ```

To open a trail file

1. Open a file with the following command:

   ```
   Logdump 6> OPEN <file_name>
   ```

   Where: <file_name> is either the relative name or fully qualified name of the file, including the file sequence number. For example:

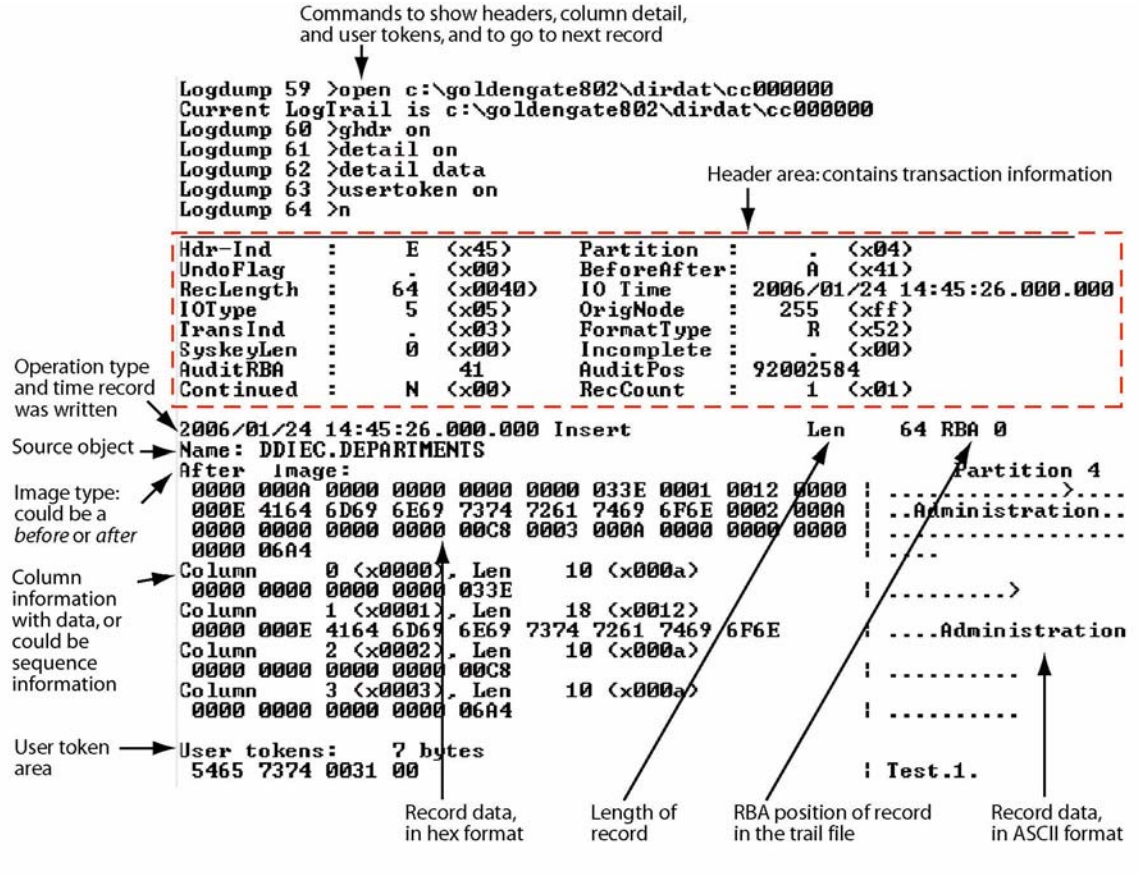
   ```
   open /home/ggs/dirdat/jd000000
   ```

2. To go to the first record and then move through records in sequence:

   ```
   Logdump 7> NEXT
   ```
   (or just type an N)

A GoldenGate trail record looks similar to the one in the following illustration, depending on what views are activated and what type of record it is. In this case, commands were issued to show the header portion of the record, to show column-level detail, and to show user tokens.

Figure 9 Sample trail record as viewed in Logdump on a Windows system



```
Logdump 59 >open c:\goldengate802\dirdat\cc000000
Current LogTrail is c:\goldengate802\dirdat\cc000000
Logdump 60 >ghdr on
Logdump 61 >detail on
Logdump 62 >detail data
Logdump 63 >usertoken on
Logdump 64 >n
___________________________________________________________________
Hdr-Ind     :     E  (x45)     Partition  :     .  (x04)
UndoFlag    :     .  (x00)     BeforeAfter:     A  (x41)
RecLength   :    64  (x0040)   IO Time    : 2006/01/24 14:45:26.000.000
IOType      :     5  (x05)     OrigNode   :   255  (xff)
TransInd    :     .  (x03)     FormatType :     R  (x52)
SyskeyLen   :     0  (x00)     Incomplete :     .  (x00)
AuditRBA    :          41      AuditPos   : 92002584
Continued   :     N  (x00)     RecCount   :     1  (x01)

2006/01/24 14:45:26.000.000 Insert               Len    64 RBA 0
Name: DDIEC.DEPARTMENTS
After  Image:                                             Partition 4
 0000 000A 0000 0000 0000 0000 033E 0001 0012 0000 | ..............>....
 000E 4164 6D69 6E69 7374 7261 7469 6F6E 0002 000A | ..Administration..
 0000 0000 0000 0000 00C8 0003 000A 0000 0000 0000 | ..................
 0000 06A4                                         | ....
Column     0 (x0000), Len    10 (x000a)
 0000 0000 0000 0000 033E                          | .........>
Column     1 (x0001), Len    18 (x0012)
 0000 000E 4164 6D69 6E69 7374 7261 7469 6F6E      | ....Administration
Column     2 (x0002), Len    10 (x000a)
 0000 0000 0000 0000 00C8                          | ..........
Column     3 (x0003), Len    10 (x000a)
 0000 0000 0000 0000 06A4                          | ..........

User tokens:    7 bytes
 5465 7374 0031 00                                 | Test.1.
```

Commands to show headers, column detail, and user tokens, and to go to next record
Header area: contains transaction information
Operation type and time record was written
Source object
Image type: could be a *before* or *after*
Column information with data, or could be sequence information
User token area
Record data, in hex format
Length of record
RBA position of record in the trail file
Record data, in ASCII format

Executing basic Logdump tasks

The following are some basic tasks that can be performed with Logdump. For detailed information about the commands shown and other available options, see the alphabetical reference beginning on page 82.

To find the next good record header

```
Logdump 8> SCANFORHEADER
```

(or just type `SFH`)

To find the beginning, middle, and end of a transaction

1. Show headers and detail.

```
Logdump 9> GHDR ON
Logdump 10> DETAIL ON
```

2. Go to the next record.

```
Logdump 11> N
```

3. View the TransInd field in the record header. The following tells you where the record is in relation to the transaction.

`TransInd : . (x00)`	First statement in transaction
`TransInd : . (x01)`	Statement in middle of transaction
`TransInd : . (x02)`	Last statement in transaction
`TransInd : . (x03)`	Sole statement in transaction

4. Move through subsequent records by pressing N, and refer to the TransInd field to determine where each one is within the transaction. When TransInd is either x02 or x03, the TransInd of the next record should be x00, starting a new transaction.

To scan for the end of a transaction

```
Logdump 20> SCANFORENDTRANSACTION
```

(or just type SFET)

The record shown will be the first one in the next transaction. To confirm, the TransInd field should be x00.

To go to a specific RBA in the file

- To go to an RBA anywhere in the file:

  ```
  Logdump 35> POS <rba>
  Logdump 36> N
  ```

  This displays the record located at that RBA.

- To go to the first record in the file:

  ```
  Logdump 37> POS FIRST
  ```

  or...

  ```
  Logdump 37> POS 0
  ```

To filter based on a table name

- To filter out everything except records containing a specific table name:

  ```
  Logdump 60> FILTER INCLUDE FILENAME <schema>.<table>
  ```

  Now, when you use the N command, you will only see records that satisfy this filter.

- Conversely, to filter out records containing a specific table name, but show everything else, use the EXCLUDE option instead of INCLUDE.

To remove the current filter criteria

```
Logdump 62> FILTER CLEAR
```

To filter on multiple conditions

```
Logdump 65> FILTER INCLUDE FILENAME <schema>.<table>; FILTER RECTYPE
<record_type>; FILTER MATCH ALL
```

Use MATCH ANY or MATCH ALL depending on whether you want the search to match any or all of the filter conditions, respectively, when multiple conditions are specified. The preceding example filters on a name and record type, typically an operation type such as INSERT.

To count the records in a trail file

```
Logdump 67> COUNT
```

This shows a count summary followed by counts for each table.

To save records to a new trail file

- To save the whole file:

  ```
  Logdump 68> SAVE <file>
  ```

  Where: <file> is the name of the new file.

- To save a subset of records:

  ```
  Logdump 69> SAVE <file> <n> RECORDS
  ```

To close the current file and open the next one in the trail

```
Logdump 70> NEXTTRAIL
```

To keep a log of your session

- To start logging:

  ```
  Logdump 71> LOG TO <filename>.txt
  ```

- To write text to the log:

  ```
  Logdump 72> WRITELOG "<text>"
  ```

- To stop logging:

  ```
  Logdump 73> LOG STOP
  ```

To see the current Logdump environment

```
Logdump 74> ENV
```

This shows which features are enabled, such as filtering and header views, and it shows environment information such as the current trail and position.

To get online command help

```
Logdump 75> HELP
```

To exit Logdump

```
Logdump 100> EXIT
```

Or…

```
Logdump 100> QUIT
```

Evaluating transaction size

Use Logdump's TRANSHIST command in conjunction with other Logdump commands to determine whether or not your applications generate large transactions and to identify their relative size. TRANSHIST causes Logdump to track the size of transactions contained in a trail file or extract file in an internal history table. The transactions are ranked in descending order of size, in bytes. When the history table is full, the smallest transaction is removed to allow a larger transaction to be added to the list.

To use statistics generated by TRANSHIST, issue the following series of commands in Logdump:

1. Use TRANSHIST to set the size of the history table that tracks transaction size. The maximum size is 200 bytes. A value of 0 turns off the tracking.

   ```
   TRANSHIST <n>
   ```

2. Use either the TRANSRECLIMIT or TRANSBYTELIMIT command to set a lower boundary for what is considered a normal sized transaction. These commands prevent normal-sized transactions from being tracked. Eliminating normal-sized transactions reduces the amount of data that must be reviewed.

   ```
   {TRANSBYTELIMIT <n bytes> | TRANSRECLIMIT <n records>}
   ```

3. Use Logdump's COUNT command to display the statistics on transaction size, which appear at the end of the output and look like the following excerpt:

   ```
       Transactions with at least 100 records or 100000 bytes
       2006/02/01 09:31:24.000.000    00:00:00.000, Seq 0, RBA 13101
           Bytes/Trans .....    1168167
           Records/Trans ...       1001
           Files/Trans .....          1
       2006/02/01 09:31:35.000.000    00:00:11.000, Seq 0, RBA 1205292
           Bytes/Trans .....    1168167
           Records/Trans ...       1001
           Files/Trans .....          1
   ```

   Logdump scans the file(s) and reports the information.

4. Use Logdump's POSITION <RBA> command to go to each RBA listed in the COUNT output to find out the name of the table that generated the transaction. You can group these tables into their own processing group so that they do not affect processing of other tables that generate normal sized transactions.

Maintaining command history

On Windows and UNIX systems, command history is stored in a file named logdump.hst. The file is created in the home location of the user who first started Logdump.

When Logdump starts up, it looks for the history file in one of the following locations, depending on the platform:

- The USERPROFILE environment variable.
- The $HOME environment variable.

If the file exists, Logdump loads the command history into a buffer. The command history buffer holds 400 commands. Upon termination of the Logdump session, the session's history is appended to the file.

Logdump command reference guide

This is the reference guide for Logdump commands.

Logdump command summary

The following are category summaries of the Logdump commands.

Table 2 Working with files

Commands	Description
CD	Sets the default directory.
LOG	Writes a session log.
NEXTTRAIL	Closes the current file and opens the next file in the trail sequence.
OPEN	Opens a trail file or extract file.
POSITION	Sets the read position in the file.
SAVE	Writes record data to another file.
WRITELOG	Writes text to a session log.

Table 3 Viewing information

Command	Description
COUNT	Displays record count information.
FILES	Displays file names in the current directory.

Table 3 Viewing information (continued)

Command	Description
ENV	Displays current Logdump settings.
FILEHEADER	Displays file header information.
NOTIFY	Displays the number of records scanned, the trail position, and the record timestamp at specified intervals when using COUNT and records are being suppressed from display through filtering options.
SHOW	Displays internal information such as the current Logdump environment, a list of GoldenGate record types, and current filter settings.
TIME	Displays the current time in local and GMT formats.

Table 4 Selecting data and records

Command	Description
DUMP	Displays the specified number of bytes of data from the current position in the file.
FILTER	Filters the display of records.
NEXT	Displays the next record(s) in the file.
SCANFORENDTRANSACTION	Finds a record that is the last record of, or the only record in, a transaction, and then displays the first record of the next transaction.
SCANFORHEADER	Finds the start of the next record header.
SCANFORRBA	Finds a specific relative byte address.
SCANFORTIME	Finds the next record with a specific timestamp.
SCANFORTYPE	Finds the next record of a specific type.
SKIP	Skips a specified number of records.

Table 5 Making conversions

Command	Description
COMPUTETIMESTAMP	Converts a datetime string to a Julian timestamp.

Table 5 Making conversions (continued)

Command	Description
CTIME	Converts a C timestamp to an ASCII timestamp.
DECRYPT	Decrypts data before displaying it in Logdump.
ENCRYPT	Encrypts file data.
INTERPRETINTERVAL	Displays a 64-bit Julian interval as days-hh:mm:ss:ms:us.
INTERPRETTIMESTAMP	Displays a 64-bit Julian timestamp in ASCII format.

Table 6 Controlling the Logdump environment

Command	Description
ASCIIDATA | EBCDICDATA	Controls whether data is displayed in ASCII or EBCDIC format on an IBM mainframe.
ASCIIDUMP | EBCDICDUMP	Sets the character set on an IBM mainframe.
ASCIIHEADER | EBCDICHEADER	Controls whether the table name is displayed in ASCII or EBCDIC format on an IBM mainframe.
DETAIL	Controls the display of detailed record information.
GHDR	Controls the display of header information.
HEADERTOKEN	Controls the display of header token indicators.
RECLEN	Sets the maximum data output length.
SCANSCROLLING	Controls whether a count notification displays on one line or multiple lines.
TIMEOFFSET	Sets the time offset from GMT.
TRAILFORMAT	Sets the trail format to the old version (pre-GoldenGate 6.0) or the new version.
TRANSBYTELIMIT	Sets a byte-count threshold for what is defined as a normal-sized transaction.
TRANSHIST	Sets the size of the transaction history table that is used for tracking transaction size.
TRANSRECLIMIT	Sets a record-count threshold for what is defined as a normal-sized transaction.

Table 6 Controlling the Logdump environment (continued)

Command	Description
USERTOKEN	Controls the display of user token data.

Table 7 Miscellaneous commands

Command	Description
DEBUG	Turns on Logdump debugging.
EXIT	Exits Logdump.
FC	Edits a previous command.
HELP	Shows syntax for Logdump commands.
HISTORY	Lists previously issued commands.
OBEY	Executes a series of commands stored in a file.
X	Executes a program from within Logdump.

ASCIIDATA | EBCDICDATA

Use ASCIIDATA or EBCDICDATA to control whether record data is displayed in ASCII or EBCDIC format on an IBM mainframe. You may need to use the ASCIIDUMP command to set the character set first.

Default `ASCIIDATA ON` (same as `EBCDICDATA OFF`)

Syntax `ASCIIDATA {ON | OFF} | EBCDICDATA {ON | OFF}`

ASCIIDUMP | EBCDICDUMP

Use ASCIIDUMP or EBCDICDUMP to set the character set for the output of the DUMP command on an IBM mainframe. The hex part of the dump display is not affected by this command.

Default `ASCIIDUMP ON` (same as `EBCDICDUMP OFF`)

Syntax `ASCIIDUMP {ON | OFF} | EBCDICDUMP {ON | OFF}`

ASCIIHEADER | EBCDICHEADER

Use ASCIIHEADER or EBCDICHEADER to control whether or not the table name in the record header is in ASCII or EBCDIC format on an IBM mainframe. You may need to use the ASCIIDUMP command to set the character set first.

Default `ASCIIHEADER ON` (same as `EBCDICHEADER OFF`)

Syntax
```
ASCIIHEADER {ON | OFF} | EBCDICHEADER {ON | OFF}
```

CD

Use CD to set the default directory. An alias for this command is VOLUME.

Default None

Syntax
```
CD <directory>
```

COMPUTETIMESTAMP

Use COMPUTETIMESTAMP to convert a datetime string to Julian format.

Default None

Syntax
```
COMPUTETIMESTAMP <datetime string>
```

Argument	Description
`<datetime string>`	A datetime string in the format of: [[yy]yy-mm-dd] [hh[:mm][:ss]]

Example
```
COMPUTETIMESTAMP 2005-01-01 12:00:00
```

This returns the following:

```
2005-01-01 12:00:00 is JulianTimestamp 211971340800000000
```

COUNT

Use COUNT to produce a record count summary and other information related to the amount of data in the file. The basic output, without options, shows the following:

- The RBA where the count began
- The number of records in the file
- The total data bytes and average bytes per record
- Information about the operation types
- Information about the transactions

When the DETAIL command is issued prior to issuing COUNT, the information includes a count for each table. COUNT options allow you to show table detail without using the DETAIL command first, set a start and end time for the count, filter the count for a table, trail file, or extract file, and specify a time interval for counts.

For arguments that take a time string, use the following format:

[[yy]yy-mm-dd] [hh[:mm][:ss]]

Default Produce a count summary of all records.

Syntax

```
COUNT
[, DETAIL]
[, END[TIME] <time_string>]
[, FILE <specification>]
[, INT[ERVAL] <minutes>]
[, LOG] <wildcard>]
[, START[TIME] <time_string>]
```

Argument	Description
`DETAIL`	Adds a count for each table that was processed by Extract to the summary count. The information includes the total and average number of data bytes and information about the operations that were performed. This data can also be obtained by using the DETAIL command before issuing COUNT.
`END[TIME] <time_string>`	Stops the count with the last record written at the specified time.
`FILE <specification>`	Specifies the count to be generated for the specified table or group of names designated with a wildcard (*).
`INT[ERVAL] <minutes>`	Displays statistics for total bytes, average bytes, and number of each type of operation that occurred within a specified interval of time, in minutes. Then it displays the totals for those statistics.
`LOG <wildcard>`	Produces a count for multiple trail or extract files specified with a wildcard.
`START[TIME] <time_string>`	Begins the count with the first record written at the specified time.

Example 1

```
COUNT START 2006-01-11 12:00:00 , END 2006-01-12 12:00:00
```

Example 2

```
COUNT INTERVAL 4
```

This displays something similar to the following (individual table count has been truncated due to space constraints):

```
Interval from 2005/02/28 11:51:18.000 to 2005/02/28 12:13:16.000,
Recs                            3
Total Data Bytes              120
  Avg Bytes/Record             40
Delete                          3
Before Images                   3
LogTrail /home/ggs/dirdat/rt000000 has 304 records
Total Data Bytes            12120
Avg Bytes/Record               39
Delete                          3
Insert                        300
FieldComp                       1
Before Images                   3
After Images                  301

Average of 303 Transactions
```

```
    Bytes/Trans .....           88
    Records/Trans ...            1
    Files/Trans .....            1
HR.JOBS                                            Partition 4
Total Data Bytes              5911
  Avg Bytes/Record              68
Insert                          86
After Images                    86

HR.REGIONS                                         Partition 4
Total Data Bytes               512
  Avg Bytes/Record              32
Insert                          16
After Images                    16
```

Example 3 `COUNT LOG ls*`

This produces a count for all files whose names begin with LS. (Individual table count has been truncated due to space constraints.)

```
Current LogTrail is c:\goldengate802\dirdat\ls000000
Bad record found at RBA 5287, format 5.50)
 2A56 623F                                         | *Vb?
LogTrail c:\goldengate802\dirdat\ls000000 has 33 records
LogTrail c:\goldengate802\dirdat\ls000000 closed
Current LogTrail is c:\goldengate802\dirdat\ls000001
LogTrail c:\goldengate802\dirdat\ls000001 has 99 records
LogTrail c:\goldengate802\dirdat\ls000001 closed
Current LogTrail is c:\goldengate802\dirdat\ls000002
LogTrail c:\goldengate802\dirdat\ls000002 has 0 records
LogTrail c:\goldengate802\dirdat\ls000002 closed
Current LogTrail is c:\goldengate802\dirdat\ls000003
LogTrail c:\goldengate802\dirdat\ls000003 has 0 records
LogTrail c:\goldengate802\dirdat\ls000003 closed
LogTrail c:\goldengate802\dirdat\ls* has 132 records

Total Data Bytes              9468
  Avg Bytes/Record              71
Insert                         132
After Images                   132

Average of 4 Transactions
    Bytes/Trans .....       3951
    Records/Trans ...         33
    Files/Trans .....          3

HR.JOBS                                            Partition 4
Total Data Bytes              5220
  Avg Bytes/Record              68
Insert                          76
After Images                    76
```

Example 4 `COUNT DETAIL`

This produces something similar to the following:

```
LogTrail /home/ggs/dirdat/rt000000 has 304 records
Total Data Bytes             12120
  Avg Bytes/Record              39
Delete                           3
Insert                         300
FieldComp                        1
Before Images                    3
After Images                   301

Average of 303 Transactions
    Bytes/Trans .....         88
    Records/Trans ...          1
    Files/Trans .....          1

GGS.TCUSTMER                                  Partition 4
Total Data Bytes             12120
  Avg Bytes/Record              39
Delete                           3
Insert                         300
FieldComp                        1
Before Images                    3
After Images                   301

Files      1, Coll      0, Chain      0
```

CTIME

Use CTIME to convert a C timestamp to an ASCII timestamp.

Default None

Syntax `CTIME <C timestamp string>`

Example `CTIME 1109823330`

This returns the following:

```
timestamp  = 1109823330 (0x42268f62)
localtime  = Wed Mar  2 20:15:30 2005
gmtime     = Thu Mar  3 04:15:30 2005
```

DEBUG

Use DEBUG to run debugging for Logdump. Use this command with the guidance of a GoldenGate support analyst.

Default Disabled

Syntax `DEBUG`

DECRYPT

Use DECRYPT to decrypt data that was encrypted with GoldenGate trail encryption, so that it can be viewed with Logdump.

Default OFF

Syntax `DECRYPT {ON | OFF}`

DETAIL

Use DETAIL to include additional information in the Logdump output. By default, Logdump only shows the hex and ASCII representation of the record.

Without options, DETAIL displays the status of record detail (ON or OFF). Options do the following:

- DETAIL ON displays a list of columns that includes the column ID, length, and value in hex and ASCII.
- DATA adds hex and ASCII data values to the column list.
- DETAIL OFF turns off detailed display.

For an illustration of how DETAIL output looks, see Figure 9 on page 78 .

DETAIL can be shortened to DET.

Default Display a column list

Syntax `DETAIL {ON | OFF | DATA}`

Argument	Description
ON	Shows detailed column information.
OFF	Suppresses detailed column information.
DATA	Adds the hex and ASCII data values to the column information.

DUMP

Use DUMP to display a HEX/ASCII or HEX/EBCDIC dump of the specified number of bytes from the open trail or extract file, starting at the current RBA.

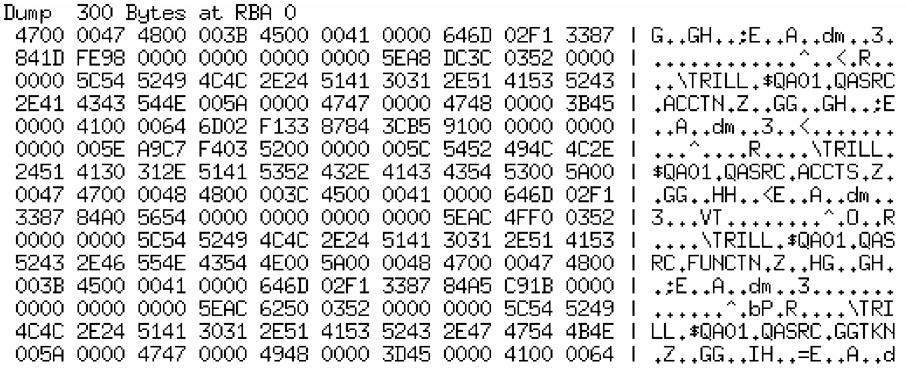
Default 256

Syntax `DUMP <bytes>`

Argument	Description
<bytes>	The number of bytes forward to display. Valid values are from 1 through 28672.

Example `DUMP 300`

This produces something similar to the following:



```
Dump  300 Bytes at RBA 0
 4700 0047 4800 003B 4500 0041 0000 646D 02F1 3387 | G..GH..;E..A..dm..3.
 841D FE98 0000 0000 0000 0000 5EA8 DC3C 0352 0000 | ............^..<.R..
 0000 5C54 5249 4C4C 2E24 5141 3031 2E51 4153 5243 | ..\TRILL.$QA01.QASRC
 2E41 4343 544E 005A 0000 4747 0000 4748 0000 3B45 | .ACCTN.Z..GG..GH..;E
 0000 4100 0064 6D02 F133 8784 3CB5 9100 0000 0000 | ..A..dm..3..<.......
 0000 005E A9C7 F403 5200 0000 005C 5452 494C 4C2E | ...^....R....\TRILL.
 2451 4130 312E 5141 5352 432E 4143 4354 5300 5A00 | $QA01.QASRC.ACCTS.Z.
 0047 4700 0048 4800 003C 4500 0041 0000 646D 02F1 | .GG..HH..<E..A..dm..
 3387 84A0 5654 0000 0000 0000 0000 5EAC 4FF0 0352 | 3...VT........^.O..R
 0000 0000 5C54 5249 4C4C 2E24 5141 3031 2E51 4153 | ....\TRILL.$QA01.QAS
 5243 2E46 554E 4354 4E00 5A00 0048 4700 0047 4800 | RC.FUNCTN.Z..HG..GH.
 003B 4500 0041 0000 646D 02F1 3387 84A5 C91B 0000 | .;E..A..dm..3.......
 0000 0000 0000 5EAC 6250 0352 0000 0000 5C54 5249 | ......^.bP.R....\TRI
 4C4C 2E24 5141 3031 2E51 4153 5243 2E47 4754 4B4E | LL.$QA01.QASRC.GGTKN
 005A 0000 4747 0000 4948 0000 3D45 0000 4100 0064 | .Z..GG..IH..=E..A..d
```

ENCRYPT

Use ENCRYPT to encrypt text supplied as an argument. The encryption method is 256-key byte substitution. The results are printed to screen.

Default None

Syntax `ENCRYPT <text>`

Example `ENCRYPT 123456789`

This produces the following:

```
Before
 3132 3334 3536 3738 39 | 123456789
After
 EF2E C1DC E4A7 68B4 14 | ......h..
```

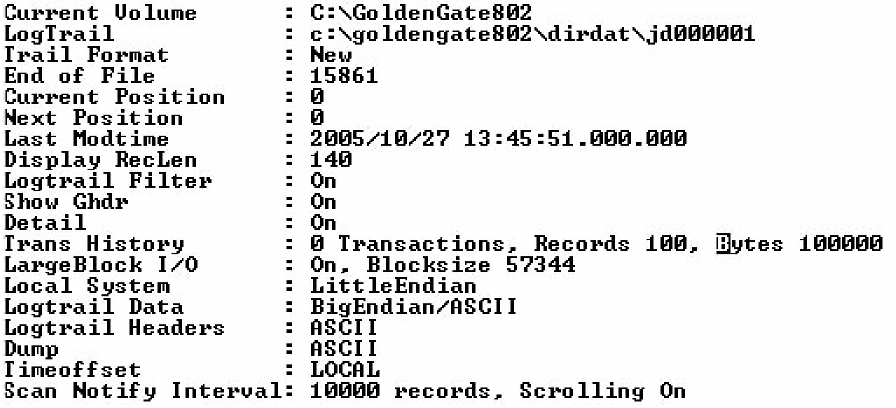
ENV

Use ENV to show current Logdump settings.

Default None

Syntax `ENV`

Example The following shows typical ENV settings.



```
Current Volume       : C:\GoldenGate802
LogTrail             : c:\goldengate802\dirdat\jd000001
Trail Format         : New
End of File          : 15861
Current Position     : 0
Next Position        : 0
Last Modtime         : 2005/10/27 13:45:51.000.000
Display RecLen       : 140
Logtrail Filter      : On
Show Ghdr            : On
Detail               : On
Trans History        : 0 Transactions, Records 100, Bytes 100000
LargeBlock I/O       : On, Blocksize 57344
Local System         : LittleEndian
Logtrail Data        : BigEndian/ASCII
Logtrail Headers     : ASCII
Dump                 : ASCII
Timeoffset           : LOCAL
Scan Notify Interval: 10000 records, Scrolling On
```

EXIT

Use EXIT to exit Logdump and terminate the process. An alias for EXIT is QUIT.

Default None

Syntax `EXIT`

FC

Use FC to edit a previously issued Logdump command and then execute it again. Previous commands are stored in the memory buffer and can be displayed by issuing the HISTORY command (see page 110). Issuing FC without arguments executes the most recently used command. By using options, you can retrieve a specific command by specifying its line number or a text substring.

Using the editor

The FC command displays the command and then opens an editor with a prompt containing a blank line starting with two dots. To edit a command, use the space bar to position the cursor beneath the character where you want the change to begin, and then enter one of the following arguments. Arguments are not case-sensitive and can be combined.

Argument	Description

`i <text>`	Inserts text. For example:
	`Logdump 24> fc 9`
	`> count`
	`..     i detail`
	`count detail`

<table>
<tr><th>Argument</th><th>Description</th></tr>
<tr><td><code>r &lt;text&gt;</code></td><td>Replaces text. For example:
<pre>
Logdump 25> fc 10
> timeoffset local
..   rgmt
timeoffset gmt
</pre></td></tr>
<tr><td><code>d</code></td><td>Deletes a character. To delete multiple characters, enter a <code>d</code> for each one. For example:
<pre>
Logdump 26> fc 11
> scanforrrbba
..   dd
scanforrba
</pre></td></tr>
<tr><td><code>&lt;replacement text&gt;</code></td><td>Replaces the displayed command with the text that you enter on a one-for-one basis. For example:
<pre>
Logdump 26> fc 10
> scanforrba 107
..   127
scanforrba 127
</pre></td></tr>
</table>

To execute the command, press Enter twice, once to exit the editor and once to issue the command. To cancel an edit, type a forward slash (/) twice.

Default Execute the most recent command again

Syntax `FC [<n> | -<n> | <string>]`

Argument	Description
`<n>`	Returns the specified command line. Each Logdump command line is sequenced, beginning with 1 at the start of the session.
`-<n>`	Returns the command that was issued `<n>` lines before the current line.
`<string>`	Returns the last command that starts with the specified text string.

Example 1 `FC 9`

Example 2 `FC -3`

Example 3 `FC sca`

FILEHEADER

Use FILEHEADER to display the contents of the header of the currently open trail file.

The file header is stored as a record at the beginning of a trail file preceding the data records. The information that is stored in the trail header provides enough information about the records to enable a GoldenGate process to determine whether the records are in a format that the current version of GoldenGate supports.

The trail header fields are stored as tokens, where the token format remains the same across all versions of GoldenGate. If a version of GoldenGate does not support any given token, that token is ignored. Depracated tokens are assigned a default value to preserve compatibility with previous versions of GoldenGate.

The current FILEHEADER command applies globally to the Logdump session, until a different FILEHEADER command is issued.

To view the file header

1. Position to the beginning of the trail file with the following Logdump command.

   ```
   pos 0
   ```

2. Issue the following Logdump command to see the first record of the file, the one that contains the file header.

   ```
   next
   ```

To retrieve the file header tokens

To retrieve file header values as input parameters, use the @GETENV function with the GGFILEHEADER option. See the *GoldenGate for Windows and UNIX Reference Guide*.

NOTE The Logdump command HEADERTOKEN also shows trail tokens, but it shows a brief summary of each one. FILEHEADER shows actual token values.

Table 8 GoldenGate file header tokens

Token/subtoken	Data Type	Description
TrailInfo		**Information about the trail file.**
Signature	UINT32	Internal use.
Compatibility	UINT16	The version of the trail. The compatibility level of the GoldenGate software must be greater than, or equal to, that of the trail file for a process to be able to read the trail file. Current valid values are 0 or 1.
CharSet	INT32	The global character set of the trail file, as defined in the parameter file or the default value. For example: WCP1252-1 -3 indicates the system default.
CreationTime	Timestamp	The time that the trail file was created, in local GMT Julian time, INT 64.

Table 8 GoldenGate file header tokens

Token/subtoken	Data Type	Description
URI	UString	The universal resource identifier of the process that created the trail file, in the format of: <host_name>:<dir>:[:<dir>][:<dir_n>]<group_name> **Where:** ◆ host_name is the name of the server that hosts the process ◆ dir is a subdirectory of the GoldenGate installation path. ◆ group_name is the name of the process group that is linked with the process. **Example**: sys1:home:oracle:v9.5:extora Shows where the trail was processed and by which process. This includes a history of previous runs.
URIHistory	UString Vector	List of the URIs of processes that wrote to the trail file before the current process. ◆ For a primary Extract, this field is empty. ◆ For a data pump, this field is URIHistory + URI of the input trail file.
FileName	UString	Name of the trail file. Can be absolute or relative path, with forward or backward slash depending on the filesystem.
MultiPart	Boolean	True/false flag indicating whether the trail file is a single file (such as one created for a batch run) or a sequentially numbered file that is part of a trail for online, continuous processing. If false, the SeqNum subtoken is not valid.
SeqNum	UINT32	The sequence number of the file in the trail, if MultiPart is true. Invalid if multipart is false. The value is the numerical sequence number, without any zero padding.
FileSize	UINT642	Size of the trail file. Value is NULL until the trail file is completed. Non-NULL values are in bytes.
FirstRecordCSN	CSN	The commit sequence number (CSN) of the first record in the trail file.Value is NULL until the trail file is completed.
LastRecordCSN	CSN	The commit sequence number (CSN) of the last record in the trail file.Value is NULL until the trail file is completed.
FirstRecordIOTime	Timestamp	The time that the first record in the trail file was written.Value is NULL until the trail file is completed.
LastRecordIOTime	Timestamp	The time that the last record in the trail file was written.Value is NULL until the trail file is completed.

Table 8 GoldenGate file header tokens

Token/subtoken	Data Type	Description
MachineInfo		**Information about the local host of the trail file.**
SysName	UString	The name of the operating system, for example: SunOX Linux Microsoft Windows
NodeName	UString	The name of the machine, for example sys1.
Release	UString	The release level of the operating system, for example: 5.10 2.6.9-11.ELsmp 2000 Advanced Server
Version	UString	The version of the operating system, for example: s10_69 #1 SMP Fri Feb 24 16:56:28 EST 2006 5.00.2195 Service Pack 4
Hardware	UString	The hardware type of the processor, for example: sun4u x86_64 x86
DatabaseInfo		**Information about the database that produced the data in the trail file.**

Vendor	UINT16	The name of the database vendor. Can be one of:
		`DB2 UDB`
		`DB2 ZOS`
		`CTREE`
		`INGRES`
		`MSSQL`
		`MYSQL`
		`ORACLE`
		`SQLMX`
		`SYBASE`
		`TERADATA`
		`TIMESTEN`
		`NONSTOP`
		`ENSCRIBE`
		`MSACCESS`
		`ODBC`

Name	UString	The name of the database, for example findb.
Instance	UString	The name of the database instance, if applicable to the database type, for example ORA1022A.

Table 8 GoldenGate file header tokens

Token/subtoken	Data Type	Description
Charset	INT32	The character set of the database. Currently, the valid value is -1 (unknown). (For some databases, this will be empty.)
MajorVersion	UINT16	The major version of the database.
MinorVersion	UINT16	The minor version of the database.
VerString	UString	The maintenance (patch) level of the database.
ClientCharset	INT32	The character set of the database client. Currently, the valid value is -1 (unknown). (For some databases, this will be empty.)
ClientVerString	UString	The maintenance (patch) level of the database client. (For some databases, this will be empty.)
ProducerInfo		**Information about the GoldenGate process that created the trail file.**
Name	UString	The group name that is associated with the process.
DataSource	UINT16	The data source that was read by the process. Can be one of: ◆ DS_EXTRACT_TRAILS (source was a GoldenGate extract file, populated with change data) ◆ DS_LOG_TABLE (source was a GoldenGate log table, used for trigger-based extraction) ◆ DS_DATABASE (source was a direct select from database table written to a trail, used for SOURCEISTABLE-driven initial load) ◆ DS_TRAN_LOGS (source was the database transaction log) ◆ DS_INITIAL_DATA_LOAD (source was Extract; data taken directly from source tables) ◆ DS_VAM_EXTRACT (source was a vendor access module) ◆ DS_VAM_TWO_PHASE_COMMIT (source was a VAM trail)
MajorVersion	UINT16	The major version of the process (xx).
MinorVersion	UINT16	The minor version of the process (xx.xx).
MaintenanceLevel	UINT16	The maintenance version of the process (xx.xx.xx).
PatchLevel	UINT16	The patch version of the process (xx.xx.xx.xx).
BuildNumber	UINT16	The build number of the process.
VerString	UString	The version string of the process. For example: 9.5.1.17A not for production

Table 8 GoldenGate file header tokens

Token/subtoken	Data Type	Description
ContinuityInfo		**Contains recovery information that is carried over from the previous trail file in the sequence.**
RecoveryMode	UINT16	Internal use
LastCompletedCSN	CSN	Internal use
LastCompletedXids	Xid	Internal use
LastSCN	CSN	Internal use
LastXid	Xid	Internal use

Default `OFF`

Syntax `FILEHEADER {ON | OFF | DETAIL}`

Argument	Description
`ON`	Enables the display of the file header, showing the main header tokens.
`OFF`	Disables the display of the file header.
`DETAIL`	Provides detailed information that includes the sub-tokens.

Example 1 `FILEHEADER ON`

```
TokenID x46 'F' Record Header    Info x00  Length  587
TokenID x30 '0' TrailInfo        Info x00  Length  303
TokenID x31 '1' MachineInfo      Info x00  Length  103
TokenID x32 '2' DatabaseInfo     Info x00  Length   88
TokenID x33 '3' ProducerInfo     Info x00  Length   85
TokenID x34 '4' ContinunityInfo  Info x00  Length    4
TokenID x5a 'Z' Record Trailer   Info x00  Length  587

2008/07/18 13:39:18.951.346 FileHeader           Len   587 RBA 0
Name: *FileHeader*
 3000 012f 3000 0008 660d 0a71 3100 0006 0001 3200 | 0../0...f..q1.....2.
 0008 0000 0016 3300 000c 02f1 7834 eac7 7f3f 3400 | ......3.....x4...?4.
 0037 0031 7572 693a 7465 6c6c 7572 6961 6e3a 3a68 | .7.1uri:tellurian::h
 6f6d 653a 6d63 6361 7267 6172 3a67 6773 3a67 6773 | ome:mccargar:ggs:ggs
 4f72 6163 6c65 3a73 6f75 7263 6536 0000 1700 112e | Oracle:source6......
 2f64 6972 6461 742f 6572 3030 3030 3030 3700 0005 | /dirdat/er0000007...
 0138 0000 0800 01e2 4039 0000 0c00 0000 0000 001d | .8......@9..........

GroupID x30 '0' TrailInfo        Info x00  Length  303
 3000 012f 3000 0008 660d 0a71 3100 0006 0001 3200 | 0../0...f..q1.....2.
 0008 0000 0016 3300 000c 02f1 7834 eac7 7f3f 3400 | ......3.....x4...?4.
 0037 0031 7572 693a 7465 6c6c 7572 6961 6e3a 3a68 | .7.1uri:tellurian::h
 6f6d 653a 6d63 6361 7267 6172 3a67 6773 3a67 6773 | ome:mccargar:ggs:ggs
 4f72 6163 6c65 3a73 6f75 7263 6536 0000 1700 112e | Oracle:source6......
 2f64 6972 6461 742f 6572 3030 3030 3030 3700 0005 | /dirdat/er0000007...
 0138 0000 0800 01e2 4039 0000 0c00 0000 0000 001d | .8......@9..........
 a33b 0000 450a 3634 3136 3138 3936 3932 0000 0000 | .;..E.6416189692....
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 3aff 0045 0000 0000 0000 | ..........:..E......
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 003d | ...................=
 0000 0c02 f178 34eb 556a 403c ff00 0c00 0000 0000 | .....x4.Uj@<........
 0000 00                                           | ...

GroupID x31 '1' MachineInfo      Info x00  Length  103
 3100 0067 3000 000b 0005 4c69 6e75 7831 0000 0f00 | 1..g0.....Linux1....
 0974 656c 6c75 7269 616e 3200 0014 000e 322e 362e | .tellurian2.....2.6.
 392d 3131 2e45 4c73 6d70 3300 0029 0023 2331 2053 | 9-11.ELsmp3..).##1 S
 4d50 2046 7269 204d 6179 2032 3020 3138 3a32 353a | MP Fri May 2018:25:
 3330 2045 4454 2032 3030 3534 0000 0c00 0678 3836 | 30 EDT 20054.....x86
 5f36 34                                           | _64

GroupID x32 '2' DatabaseInfo     Info x00  Length   88
 3200 0058 3000 0006 0007 3100 000e 0008 4f52 4131 | 2..X0.....1.....ORA1
 3032 3241 3200 000e 0008 6f72 6131 3032 3261 3300 | 022A2.....ora1022a3.
 0008 ffff ffff 3400 0006 0000 3500 0006 0000 3600 | ......4.....5.....6.
 0006 0000 3700 0008 ffff ffff 3800 0010 000a 3130 | ....7.......8.....10
 2e32 2e30 2e32 2e30                               | .2.0.2.0

GroupID x33 '3' ProducerInfo     Info x00  Length   85
 3300 0055 3000 000a 0004 4546 4152 3100 0006 0003 | 3..U0.....EFAR1.....
 3200 0006 0000 3300 0006 0000 3400 0006 0000 3500 | 2.....3.....4.....5.
 0006 0000 3600 0006 0017 3700 0023 001d 5665 7273 | ....6.....7..#..Vers
 696f 6e20 5374 2e20 416e 6472 6577 7320 4275 696c | ion St. Andrews Buil
 6420 3032 33                                      | d 023

GroupID x34 '4' ContinunityInfo  Info x00  Length    4
 3400 0004                                         | 4...
```

Example 2

```
FILEHEADER DETAIL
TokenID x46 'F' Record Header    Info x00  Length  587
TokenID x30 '0' TrailInfo        Info x00  Length  303
TokenID x31 '1' MachineInfo      Info x00  Length  103
TokenID x32 '2' DatabaseInfo     Info x00  Length   88
TokenID x33 '3' ProducerInfo     Info x00  Length   85
TokenID x34 '4' ContinunityInfo  Info x00  Length    4
TokenID x5a 'Z' Record Trailer   Info x00  Length  587

2008/07/18 13:40:26.034.631 FileHeader           Len   587 RBA 0
Name: *FileHeader*
```

```
 3000 012f 3000 0008 660d 0a71 3100 0006 0001 3200 | 0../0...f..q1.....2.
 0008 0000 0016 3300 000c 02f1 7834 eac7 7f3f 3400 | ......3.....x4...?4.
 0037 0031 7572 693a 7465 6c6c 7572 6961 6e3a 3a68 | .7.1uri:tellurian::h
 6f6d 653a 6d63 6361 7267 6172 3a67 6773 3a67 6773 | ome:mccargar:ggs:ggs
 4f72 6163 6c65 3a73 6f75 7263 6536 0000 1700 112e | Oracle:source6......
 2f64 6972 6461 742f 6572 3030 3030 3030 3700 0005 | /dirdat/er0000007...
 0138 0000 0800 01e2 4039 0000 0c00 0000 0000 001d | .8......@9..........

GroupID x30 '0' TrailInfo        Info x00  Length  303
 3000 012f 3000 0008 660d 0a71 3100 0006 0001 3200 | 0../0...f..q1.....2.
 0008 0000 0016 3300 000c 02f1 7834 eac7 7f3f 3400 | ......3.....x4...?4.
 0037 0031 7572 693a 7465 6c6c 7572 6961 6e3a 3a68 | .7.1uri:tellurian::h
 6f6d 653a 6d63 6361 7267 6172 3a67 6773 3a67 6773 | ome:mccargar:ggs:ggs
 4f72 6163 6c65 3a73 6f75 7263 6536 0000 1700 112e | Oracle:source6......
 2f64 6972 6461 742f 6572 3030 3030 3030 3700 0005 | /dirdat/er0000007...
 0138 0000 0800 01e2 4039 0000 0c00 0000 0000 001d | .8......@9..........
 a33b 0000 450a 3634 3136 3138 3936 3932 0000 0000 | .;..E.6416189692....
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 3aff 0045 0000 0000 0000 | ..........:..E......
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 003d | ...................=
 0000 0c02 f178 34eb 556a 403c ff00 0c00 0000 0000 | .....x4.Uj@<........
 0000 00                                           | ...
TokenID x30 '0' Signature        Info x00  Length    8
 660d 0a71                                         | f..q
TokenID x31 '1' Compatibility    Info x00  Length    6
 0001                                              | ..
TokenID x32 '2' Charset          Info x00  Length    8
 0000 0016                                         | ....
TokenID x33 '3' CreationTime     Info x00  Length   12
 02f1 7834 eac7 7f3f                               | ..x4...?
TokenID x34 '4' URI              Info x00  Length   55
 0031 7572 693a 7465 6c6c 7572 6961 6e3a 3a68 6f6d | .1uri:tellurian::hom
 653a 6d63 6361 7267 6172 3a67 6773 3a67 6773 4f72 | e:mccargar:ggs:ggsOr
 6163 6c65 3a73 6f75 7263 65                       | acle:source
TokenID x36 '6' Filename         Info x00  Length   23
 0011 2e2f 6469 7264 6174 2f65 7230 3030 3030 30   | .../dirdat/er000000
TokenID x37 '7' MultiPart        Info x00  Length    5
 01                                                | .
TokenID x38 '8' Seqno            Info x00  Length    8
 0001 e240                                         | ...@
TokenID x39 '9' FileSize         Info x00  Length   12
 0000 0000 0000 1da3                               | ........
TokenID x3b ';' LastCSN          Info x00  Length   69
 0a36 3431 3631 3839 3639 3200 0000 0000 0000 0000 | .6416189692.........
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 00                                      | .....
TokenID x3a ':' FirstCSN         Info xff  Length   69
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 | ....................
 0000 0000 00                                      | .....
TokenID x3d '=' LastIOTime       Info x00  Length   12
 02f1 7834 eb55 6a40                               | ..x4.Uj@
TokenID x3c '<' FirstIOTime      Info xff  Length   12
 0000 0000 0000 0000                               | ........

GroupID x31 '1' MachineInfo      Info x00  Length  103
 3100 0067 3000 000b 0005 4c69 6e75 7831 0000 0f00 | 1..g0.....Linux1....
 0974 656c 6c75 7269 616e 3200 0014 000e 322e 362e | .tellurian2.....2.6.
 392d 3131 2e45 4c73 6d70 3300 0029 0023 2331 2053 | 9-11.ELsmp3..).##1 S
 4d50 2046 7269 204d 6179 2032 3020 3138 3a32 353a | MP Fri May 20 18:25:
 3330 2045 4454 2032 3030 3534 0000 0c00 0678 3836 | 30 EDT 20054.....x86
 5f36 34                                           | _64
TokenID x30 '0' Sysname          Info x00  Length   11
 0005 4c69 6e75 78                                 | ..Linux
TokenID x31 '1' Nodename         Info x00  Length   15
 0009 7465 6c6c 7572 6961 6e                       | ..tellurian
TokenID x32 '2' Release          Info x00  Length   20
 000e 322e 362e 392d 3131 2e45 4c73 6d70           | ..2.6.9-11.ELsmp
TokenID x33 '3' Version          Info x00  Length   41
 0023 2331 2053 4d50 2046 7269 204d 6179 2032 3020 | .##1 SMP Fri May 20
 3138 3a32 353a 3330 2045 4454 2032 3030 35        | 18:25:30 EDT 2005
TokenID x34 '4' Hardware         Info x00  Length   12
 0006 7838 365f 3634                               | ..x86_64

GroupID x32 '2' DatabaseInfo     Info x00  Length   88
 3200 0058 3000 0006 0007 3100 000e 0008 4f52 4131 |2..X0.....1.....ORA1
 3032 3241 3200 000e 0008 6f72 6131 3032 3261 3300 |022A2.....ora1022a3.
 0008 ffff ffff 3400 0006 0000 3500 0006 0000 3600 |......4.....5.....6.
 0006 0000 3700 0008 ffff ffff 3800 0010 000a 3130 |....7.......8.....10
 2e32 2e30 2e32 2e30                               | .2.0.2.0
TokenID x30 '0' Vendor           Info x00  Length    6
 0007                                              | ..
TokenID x31 '1' Name             Info x00  Length   14
 0008 4f52 4131 3032 3241                          | ..ORA1022A
TokenID x32 '2' Instance         Info x00  Length   14
 0008 6f72 6131 3032 3261                          | ..ora1022a
TokenID x33 '3' Charset          Info x00  Length    8
 ffff ffff                                         | ....
TokenID x34 '4' MajorVersion     Info x00  Length    6
 0000                                              | ..
TokenID x35 '5' MinorVersion     Info x00  Length    6
 0000                                              | ..
TokenID x36 '6' VerString        Info x00  Length    6
 0000                                              | ..
TokenID x37 '7' ClientCharset    Info x00  Length    8
 ffff ffff                                         | ....
TokenID x38 '8' ClientVerString  Info x00  Length   16
 000a 3130 2e32 2e30 2e32 2e30                     | ..10.2.0.2.0
```

```
GroupID x33 '3' ProducerInfo     Info x00  Length   85
 3300 0055 3000 000a 0004 4546 4152 3100 0006 0003 | 3..U0.....EFAR1.....
 3200 0006 0000 3300 0006 0000 3400 0006 0000 3500 | 2.....3.....4.....5.
 0006 0000 3600 0006 0017 3700 0023 001d 5665 7273 | ....6.....7..#..Vers
 696f 6e20 5374 2e20 416e 6472 6577 7320 4275 696c | ion St. Andrews Buil
 6420 3032 33                                      | d 023
TokenID x30 '0' Name             Info x00  Length   10
 0004 4546 4152                                    | ..EFAR
TokenID x31 '1' DataSource       Info x00  Length    6
 0003                                              | ..
TokenID x32 '2' MajorVersion     Info x00  Length    6
 0000                                              | ..
TokenID x33 '3' MinorVersion     Info x00  Length    6
 0000                                              | ..
TokenID x34 '4' MaintLevel       Info x00  Length    6
 0000                                              | ..
TokenID x35 '5' BugFixLevel      Info x00  Length    6
 0000                                              | ..
TokenID x36 '6' BuildNumber      Info x00  Length    6
 0017                                              | ..
TokenID x37 '7' VerString        Info x00  Length   35
 001d 5665 7273 696f 6e20 5374 2e20 416e 6472 6577 | ..Version St.Andrew
 7320 4275 696c 6420 3032 33                       | s Build 023

GroupID x34 '4' ContinunityInfo  Info x00  Length    4
 3400 0004                                         | 4...
```

FILES

Use FILES to display summary file information for files on the local system. The default command displays all files in the curent directory. To constrain the display to specific files, you can supply a wildcarded name.

This command can be shortened to FI. An alias for this command is DIR or FILEINFO.

Default Show all files in current directory

Syntax `FILES [<directory>]`

Argument	Description
`<directory>`	The name of a directory or a wildcard for specific files. Note: On a Windows system, if any file or directory in the specified path contains spaces, the entire path must be enclosed within double quotation marks.

Example `FILES "c:\goldengate ver802\dirdat\cc*"`

FILTER

Use FILTER to filter the display based on one or more criteria.

- You can string multiple FILTER commands together, separating each one with a semicolon, as in:

  ```
  FILTER INCLUDE FILENAME fin.act*; FILTER RECTYPE 5; FILTER MATCH ALL
  ```

- To avoid unexpected results, avoid stringing filter options together with one FILTER command. For example, the following would be *incorrect*:

  ```
  FILTER INCLUDE FILENAME fin.act*; RECTYPE 5; MATCH ALL
  ```

Without arguments, FILTER displays the current filter status (ON or OFF) and any filter criteria that are in effect.

Comparison operators

For options that take comparison operators, standard operators may be used. These are:

Table 9 Filter option comparison operators

Operator	Example
Equal	= EQ ==
Less than	< LT
Less than or equal	<= LE
Greater than	> GT
Greater than or equal	>= GE
Not equal	<> NE !=

Note: The absence of an operator implies Equal.

Default Shows current filter settings

Syntax `FILTER [INCLUDE] [EXCLUDE] <filter option>`

Where:

<filter option> can be one of:

```
{
AUDITRBA <rba> [<comparison operator>] |
CLEAR {<filter_spec> | ALL} |
CSN | LogCSN [<comparison operator>] [<value>]
ENDTIME <time_string> |
FILENAME <name> [, <name>] |
GGSTOKEN <token name> [<comparison operator>] [<token value>] |
HEX “<hex_string>” [<byte_range>][, “<hex_string>” [<byte_range>]] [...]
|
INT16 <16-bit_integer> |
INT32 <32-bit_integer> |
IOTYPE <operation type> [, <operation type>] |
MATCH {ANY | ALL} |
OFF |
ON |
PROCESS <process_name> |
RBA <byte address> [<comparison operator>] [...] |
RECLEN <length> [<comparison operator>] |
RECTYPE {<type_number> | <type_name>} |
SHOW |
STARTTIME <time_string> |
STRING [BOTH] [B],<text> [<column_range>]
   [[B],<text> [<column_range>]] [...] |
SYSKEY <system key> [<comparison operator>] [...] |
TRANSIND <indicator> [<comparison operator>] |
TYPE <type> |
UNDOFLAG <type> [<comparison operator>] |
USERTOKEN <token name> [<comparison operator>] [<token value>]
}
```

Argument	Description

`AUDITRBA <rba>`
`[<comparison operator>]`
	Filters based on the relative byte address of a commit record. For <comparison operator>, see "Comparison operators" on page 103.

`CLEAR {<filter_spec> | ALL}`	Removes filtering criteria.
	- ALL removes all filter criteria.
	- <filter_spec> removes only the specified criterion. Specify any FILTER option, but not the value. The following example is valid:
	`FILTER CLEAR STRING`
	The following example is not valid:
	`FILTER CLEAR STRING "Denver"`
	An alias for CLEAR is RESET.

`CSN | LogCSN [<comparison operator>] [<value>]`	Filters based on a commit sequence value. For <comparison operator>, see "Comparison operators" on page 103.
`ENDTIME <time_string>`	Ends the filter at the last record written at the specified time. For the time string, use the format of: [[yy]yy-mm-dd] [hh[:mm][:ss]] Example: `ENDTIME 2005-01-12 00:00:00` Can be shortened to ENDTS or END.

`FILENAME <name>`
`[, <name>]`
	Filters based on the name of a SQL table, or a group of names, with the name format being:
	[catalog.]<owner>.<table>
	[catalog.]<owner>.<string>*
	Also filters on the name of a NonStop data file, or a group of names, with the name format being:
	<volume>.<subvolume>.<file>
	<volume>.<subvolume>.<string>*
	FILENAME is case-sensitive. FILENAME can be shortened to FILE or FI.
	Up to eight name specifications may be supplied.

Argument	Description
`HEX "<hex_string>" [<byte_range>] [, "<hex_string>" [<byte_range>]] [...]`	Filters based on a hex string and, optionally, a range of columns. To specify a range of columns, use the format of: <start_column>:<end_column> Example: `10:35` This option allows up to eight hex string and column arguments. Hex strings must be enclosed within quotes.
`INCLUDE`	Specifies that the filter will include the information specified with other options in the current FILTER statement. Can be shortened to INC.
`EXCLUDE`	Specifies that the filter will exclude the information specified with other options in the current FILTER statement. Can be shortened to EXC.
`INT16 <16-bit_integer>`	Filters based on a 16-bit integer. Use with 16-bit processors.
`INT32 <32-bit_integer>`	Filters based on a 32-bit integer. Use with 32-bit processors.
`IOTYPE <operation type> [, <operation type>]`	Filters based on the type of operation. A list of record types can be viewed with the SHOW RECTYPE command in Logdump. Up to 32 operation types can be specified with IOTYPE.
`MATCH {ANY | ALL}`	Controls filtering response when multiple filters have been specified. Can be shortened to MAT or MA. ◆ ANY includes a record for display or counts if the condition matches any of the filter conditions. This is the default. ◆ ALL includes a record for display or counts only if the condition matches all of the filter conditions.
`OFF`	Disables record filtering. By default, filtering is disabled. An alias for this option is DISABLE.
`ON`	Enables record filtering. An alias for this option is ENABLE.
`RBA <byte address> [<comparison operator>] [...]`	Filters based on a relative byte address. Accepts either a 32-bit or 64-bit value. Up to 32 specifications can be supplied.
`RECLEN <length> [<comparison operator>]`	Filters based on a record length, in bytes. For <comparison operator>, see "Comparison operators" on page 103.

Argument	Description
RECTYPE {<type_number> | <type_name>}	Filters based on the type of record. Can be either of the following: ◆ The number assigned to the record type. FILTER RECTYPE 10 ◆ The name of the record type. FILTER RECTYPE Update To view the record type names and numbers, issue the SHOW RECTYPE command. (See page 118.)
SHOW	Displays filter settings. Same as using FILTER without any options.
STARTTIME <time_string>	Starts the filter with the first record written at the specified time. For the time string, use the format of: [[yy]yy-mm-dd] [hh[:mm][:ss]] Example: STARTTIME 2005-01-11 00:00:00 Can be shortened to STARTTS or START.
STRING [BOTH] [B], <text> [<column_range>] [[B], <text> [<column_range>]] [...]	◆ <text> filters based on a string. Enclose the string within quotes. ◆ <column_range> filters based on a range of columns. Use the format of: <start_column>:<end_column> Example: 10:35 ◆ BOTH filters on both a string and a column range. ◆ [B] specifies a case-insensitive match. You can match up to eight string and column arguments. If the trail data is EBCDIC, issue the EBCDICDATA ON or ASCIIDATA OFF command before using FILTER STRING to ensure the correct matching. STRING can be shortened to STR.
SYSKEY <system key> [<comparison operator>] [...]	Filters based on a NonStop source key. Accepts either a 32-bit or 64-bit value. Up to 32 specifications can be supplied.

Argument	Description

`TRANSIND <indicator>`
`[<comparison operator>]`
	Filters based on the TransInd field of the record header. Valid values:
	0 = start of transaction
	1 = middle of transaction
	2 = end of transaction
	3 = only record in transaction
	For example, to filter for the end of a transaction, use the following command, including the spaces in the syntax:
	`FILTER INCLUDE TransInd > = 2`
	For <comparison operator>, see "Comparison operators" on page 103.

`UNDOFLAG <type>`
`[<comparison operator>]`
	Filters based on the NonStop undo flag. The undo flag is set for records that are undone when a TMF transaction is aborted. Normally, UndoFlag is set to zero, but if the record is the backout of a previously successful operation, then UndoFlag will be set to 1. An undo that is performed by the disc process because of a constraint violation is not marked as an undo.
	For <comparison operator>, see "Comparison operators" on page 103.

`USERTOKEN <token name>`
`[<comparison operator>]`
`[<token value>]`
	Filters based on a specific user token in the trail file header.
	◆ <token name> is the name of any token that is defined with the TOKENS clause of a TABLE statement of the Extract parameter file. It is not case-sensitive.
	◆ <token value> is either a constant that is enclosed within double quotes or the result of a GoldenGate column-conversion function, depending on what was specified in the TOKENS clause for <token name>.
	◆ For <comparison operator>, see "Comparison operators" on page 103.

Example 1 The following shows filter options modified by comparison operators.

```
FILTER INCLUDE RECLEN > 400
FILTER INCLUDE RECLEN < 200
FILTER INCLUDE TRANSIND <> 1
FILTER INCLUDE SYSKEY > 202172700557313
```

Example 2 The following filters for a data file name and for a relative key 19446, which has a hex value of 00004bf6. Because MATCH ALL is used, a record must meet all of the filter specifications to be included in the filter.

```
FILTER INCLUDE FILENAME $QA01.QAESRC.ACCT*
FILTER INCLUDE HEX “00004bf6” 0:3
FILTER MATCH ALL
```

Example 3 The following filters for tables that start with "ACC," except for the ACCDET table, and for records that contain a timestamp between the specified start and stop times. By default, if a record matches any of the INCLUDE specifications, it is included in the filter.

```
FILTER INCLUDE FILENAME SALES.ACC*
FILTER EXCLUDE FILENAME SALES.ACCDET
FILTER INCLUDE STARTTIME 2005-01-11 17:30:15
FILTER INCLUDE ENDTIME 2005-01-11 19:00:00
```

Example 4 The following shows filter options with multiple specifications. By default, a record that matches any of these specifications will be included in the filter. Note that in the STRING filter, two of the criteria are not case-sensitive, while one is, and the filter is confined to a column range.

```
FILTER INCLUDE IOTYPE insert,update,delete
FILTER INCLUDE STRING b”String1” “string2” b“String3” 25:50
FILTER INCLUDE FILENAME $QA01.QAESRC.ACCT1, $QA01.QAESRC.ACCT2,
$QA01.QAESRC.ACCT3
```

GHDR

Use GHDR to control whether or not the record header is displayed with each record. Each record contains a header that includes information about the transaction environment. Without arguments, GHDR displays the status of header display (ON or OFF).

Default OFF

Syntax GHDR {ON | OFF}

HEADERTOKEN

Use HEADERTOKEN to control whether or not header token indicators are displayed with each record. The header token indicators are the following:

G — record header (begin of record)

H — header area

D — data area

T — GoldenGate internal token

U — user token area (does not display if user tokens are not in use)

Z — end of record

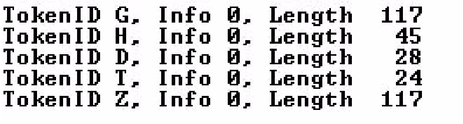
Without arguments, HEADERTOKEN displays the status of header token indicators (ON or OFF).

Default OFF

Syntax `HEADERTOKEN {ON | OFF | DETAIL}`

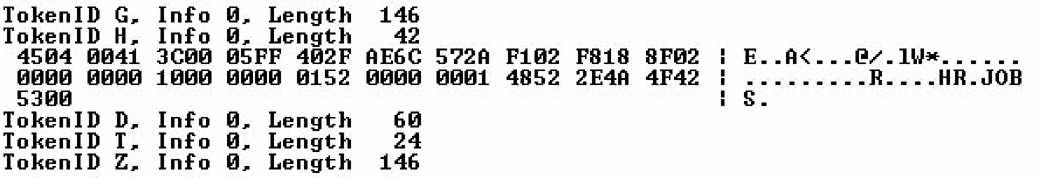
Argument	Description
ON	Enables the display of header tokens.
OFF	Disables the display of header tokens.
DETAIL	Provides detailed token values.

Example 1 HEADERTOKEN, without DETAIL



```
TokenID G, Info 0, Length  117
TokenID H, Info 0, Length   45
TokenID D, Info 0, Length   28
TokenID T, Info 0, Length   24
TokenID Z, Info 0, Length  117
```

Example 2 HEADERTOKEN with DETAIL



```
TokenID G, Info 0, Length  146
TokenID H, Info 0, Length   42
 4504 0041 3C00 05FF 402F AE6C 572A F102 F818 8F02 | E..A<...@/.lW*......
 0000 0000 1000 0000 0152 0000 0001 4852 2E4A 4F42 | .........R....HR.JOB
 5300                                              | S.
TokenID D, Info 0, Length   60
TokenID T, Info 0, Length   24
TokenID Z, Info 0, Length  146
```

HELP

Use HELP to view the syntax of Logdump commands.

Default None

Syntax `HELP`

HISTORY

Use HISTORY to view the most recently issued Logdump commands since the session started, or to reset the command count starting at line 1 again. HISTORY can be shortened to HIST.

NOTE You can use the FC command to re-execute a command in the list. See page 92.

Default Display recent commands

Syntax `HISTORY [<n>] [CLEAR]`

Argument	Description
<n>	Returns the specified number of previously issued commands, where <n> is any positive number.

Argument	Description
CLEAR	Deletes the command history buffer and reverts the command line to 1.

Example

```
HISTORY 3
```

The results of this command would be similar to:

```
1: ghdr on
2: detail on
3: scanforheader
```

INTERPRETINTERVAL

Use INTERPRETINTERVAL to display a 64-bit Julian time interval in the format of days-hh:mm:ss.ms.us.

Default None

Syntax `INTERPRETINTERVAL <interval string>`

Argument	Description
<interval string>	A string representing the interval to be converted.

Example

```
INTERPRETINTERVAL 1234567
```

This produces the following result:

```
Interval 1234567 is 0-00:00:01.234.567
```

INTERPRETTIMESTAMP

Use INTERPRETTIMESTAMP to display a 64-bit Julian timestamp as an ASCII value.

Default None

Syntax `INTERPRETTIMESTAMP <timestamp>`

Argument	Description
<timestamp>	A JULIANTIMESTAMP value.

Example

```
INTERPRETTIMESTAMP 211976584185800569
```

This produces the following result:

```
2005/03/03 04:29:45.800.569 GMT
2005/03/02 20:29:45.800.569 LCT
```

LOG

Use LOG to start and stop the logging of Logdump sessions. When enabled, logging remains in effect for all sessions of Logdump until disabled with the LOG STOP command. Without arguments, LOG displays the status of logging (ON or OFF). An alias for LOG is OUT.

Default Disabled

Syntax `LOG <file_name> | STOP}`

Argument	Description
`<file_name>`	Specifies the name of the log file. Specify a full path name to store the file in a directory other than the current working directory.
`STOP`	Stops logging.

Example `LOG /home/ggs/dirrpt/logdumpout.txt`

NEXT

Use NEXT to display the next record or records in the file. The default displays only the next record. NEXT can be shortened to N. An alias for NEXT is RECORD.

Default Display the next 1 record

Syntax `NEXT [<n>]`

Argument	Description
`<n>`	Displays the specified number of subsequent records.

Example `NEXT 10`

NEXTTRAIL

Use NEXTTRAIL to close an open trail file and open the next one in the sequence. An alias for NEXTTRAIL is NT.

Default None

Syntax `NEXTTRAIL`

NOTIFY

Use NOTIFY to display the number of records scanned, the trail position, and the record timestamp at specified intervals when using COUNT and records are being suppressed from display through filtering options. An alias for NOTIFY is NOTIFYINTERVAL.

Instead of displaying each notify interval on a separate line, you can configure Logdump to simply update a single line with each new scan result. See “SCANSCROLLING” on page 118.

Default None

Syntax `NOTIFY <interval>`

Argument	Description
`<interval>`	The notification interval expressed as a number of records.

Example The following shows the usage and result of this command.

```
Logdump 26> NOTIFY 1000
Logdump 27> FILTER INCLUDE FILE sales.res*
Logdump 28> COUNT
Scanned 1000 records, RBA 160380,2005/02/24 08:53:47.768.255
Scanned 2000 records, RBA 729961,2005/02/24 08:56:09.916.128
Scanned 3000 records, RBA 2032683,2005/02/24 08:56:09.916.128
Scanned 4000 records, RBA 3244585,2005/02/24 08:56:09.916.128
Scanned 5000 records, RBA 4568766,2005/02/24 08:56:09.916.128
```

OBEY

Use OBEY to process a file that contains a list of Logdump commands. OBEY is useful for executing commands that are frequently used in sequence.

OBEY can be shortened to O. An alias for OBEY is SOURCE.

Default None

Syntax `OBEY <file name>`

Argument	Description
`<file name>`	The fully qualified name of the file containing the list of commands.

Example `OBEY ./ldcommands.txt`

The preceding command executes a file that might look something like this:

```
ghdr on
usertoken on
detail
filter enable
filter clear
filter match all
```

OPEN

Use OPEN to open a trail file or extract file in Logdump. Without arguments, the command displays the name of the file that is currently open. Aliases for OPEN are FROM and LOGTRAIL.

Default None

Syntax `OPEN <file_name>`

Argument	Description
`<file_name>`	The fully qualified path name of the trail file or extract file to be opened. To specify a trail file, specify the trail name (a two-character prefix) and the sequence number, for example jd000000.

Example `OPEN /home/ggs/dirdat/jd000000`

POSITION

Use POSITION to set the read position in the file. The position of a record in the file is noted in the record header in the AuditPos field.

Without options, POSITION displays the current read position. Options let you specify an exact position. After you set the position, issue the NEXT command to view the record at that position.

POSITION can be shortened to POS.

Default None

Syntax `POSITION [<bytes> | {0 | FIRST}]`

Argument	Description
`<bytes>`	Specifies the number of bytes into the file at which to read. Use the NEXT command to view the specified record.
`0 | FIRST`	Positions Logdump at the beginning of the file.

Example `POS 77580548`

RECLEN

Use RECLEN to control how much of the record data is displayed. You can use RECLEN to control the amount of scrolling that must be done when records are large, while still showing enough data to evaluate the record. Data beyond the specified length is truncated.

Default 140 bytes

Syntax `RECLEN <n>`

Argument	Description
`<n>`	The number of bytes of the record that is displayed.

Example `RECLEN 280`

SAVE

Use SAVE to write a subset of the records to a new trail or extract file. By saving a subset to a new file, you can work with a smaller file that is easier to debug. Saving to another file also enables you to extract valid records that can be processed by GoldenGate, while excluding records that may be causing errors.

To set the version of the trail or file (to old or new format), use the TRAILFORMAT command.

Default None

```
SAVE <file_name> [!] {<n> records | <n> bytes}[NOCOMMENT]
[OLDFORMAT | NEWFORMAT]
[TRANSIND <indicator>]
[TRUNCATE]
```

Argument	Description
`<file_name>`	The name of the new file. To specify a trail file, specify the two-character trail name and a sequence number, for example rt000001.
`!`	Overwrites the specified file, if the same file already exists. First a purge is done, and then the specified records are saved to the file.

`<n> records |`
`<n> bytes`
	Specifies either a number of records or a number of data bytes to write to the new file. The <n> number of records or bytes are taken forward from the current position in the file. You can change the position with the POSITION command. See page 114.

`NOCOMMENT`	Suppresses the leading and trailing comment records that are placed by default in the new file. These records describe the context of the file. The begin comment record contains source trail information and the position where the save started. The end comment record identifies the end of the saved data. These headers are useful to separate different sets of records that are saved to the same file, but can be omitted.

`OLDFORMAT |`
`NEWFORMAT`
	Writes the data in either the current trail format (NEWFORMAT, the default) or the format that was used for GoldenGate versions 6.0 and earlier (OLDFORMAT).

`TRANSIND`
`<indicator>`
	Sets the TransInd header field in the records written to one of the following:
	`FIRST`
	`MIDDLE`
	`END`
	`ONLY`
	This allows you to reorder records in a transaction. TRANSIND applies to all records written by a SAVE command.

`TRUNCATE`	Purges an existing file before saving new information to it.

Example

```
SAVE /home/ggs/dirdat/rt000001 10 records nocomment
```

SCANFORENDTRANSACTION

Use SCANFORENDTRANSACTION to scan for a record that has a transaction indicator of 2 or 3, as shown in the TransInd field of the header. When one of those indicators is found, Logdump displays the first record of the next transaction.

The indicators represent the following:

- 2 — last record in the transaction
- 3 — only record in the transaction

SCANFORENDTRANSACTION can be shortened to SFET.

Default None

Syntax `SCANFORENDTRANSACTION`

SCANFORHEADER

Use SCANFORHEADER to go to the next record header. Before using this command, use the GHDR ON command to show record headers (see page 109). SCANFORHEADER can be shortened to SFH.

Default None

Syntax `SCANFORHEADER [PREV]`

Argument	Description
PREV	Displays the previous record header.

SCANFORRBA

Use SCANFORRBA to scan for the record at a relative byte address specified by the AuditRBA field of the record header. Before using this command, use the GHDR command to show record headers (see page 109). SCANFORRBA can be shortened to SFR.

Default None

Syntax `SCANFORRBA <relative byte address> [<file_name>]`

Argument	Description
`<relative byte address>`	Specifies the relative byte address to find.
`<file_name>`	Constrains the search to an Enscribe or SQL data file. A file name is required even if you are searching a file that is open in Logdump.

Example SCANFORRBA `321 /home/ggs/dirdat/rt000000`

SCANFORTIME

Use SCANFORTIME to scan for a record that contains a specific timestamp. The timestamp is contained in the IO Time field of the record header. Before using this command, use the GHDR command to show record headers (see page 109). SCANFORTIME can be shortened to SFTS.

Default None

Syntax `SCANFORTIME <time_string> [, <name>]`

Argument	Description
`<time_string>`	Scans for a specific timestamp. For the time string, use the format of: [[yy]yy-mm-dd] [hh[:mm][:ss]]
`<name>`	Constrains the search to a specific table name, or a group of names specified with a wildcard.

Example SCANFORTIME `2005-10-27 14:33:57`

SCANFORTYPE

Use SCANFORTYPE to scan for the next record of the specified type. SCANFORTYPE can be shortened to SFT.

Default None

Syntax `SCANFORTYPE {<type_name> | <type_number>}`

Argument	Description
`<type_name> | <type_number>`	Specifies the type of record to search for, either by type name or type number. To view a list of record types and their associated numbers, use the SHOW RECTYPE command (see page 118).

Example Both of the following commands return the same result: They display commit records.

```
SCANFORTYPE Commit
SFT 2
```

SCANSCROLLING

Use SCANSCROLLING to configure Logdump to update a single line after COUNT scans when NOTIFY is enabled. Otherwise, each scan notification appears on a different line. See "NOTIFY" on page 112 for more information.

Default OFF

Syntax `SCANSCROLLING {ON | OFF}`

Argument	Description
ON	Enables the use of a single line for count notification results.
OFF	Disables the use of a single line, causing a separate line to be used for each notification.

SHOW

Use SHOW to display internal Logdump information, including the current Logdump environment, a list of GoldenGate record types, and current filter settings. SHOW can be shortened to SH or SHO.

Default None

Syntax

```
SHOW
[ENV]
[FILTER]
[RECTYPE]
```

Argument	Description
ENV	Displays the current Logdump environment. Same as the ENV command (see page 91).
FILTER	Displays current filter settings.
RECTYPE	Displays a list of GoldenGate record types that can be displayed with Logdump.

Example 1 `SHOW FILTER`

This shows something similar to the following:

```
Data filters are ENABLED
Include  Match ALL
Rectypes   : Delete
Filename-0 : hr.regions
Exclude  Match ANY
```

Example 2 `SHOW RECTYPE`

This shows results similar to the following. (This list might not reflect all possible record types. New types are added when needed to support new functionality.)

```
LogTrail record types
   1 - Abort                             2 - Commit
   3 - Delete                            4 - EndRollBack
   5 - Insert                            6 - Prepared
   7 - TMF-Shutdown                      8 - TransBegin
   9 - TransRelease                     10 - Update
  11 - UpdateComp                       12 - FileAlter
  13 - FileCreate                       14 - FilePurge
  15 - FieldComp                        16 - FileRename
  17 - AuxPointer                       18 - NetworkCommit
  19 - NetworkAbort                     20 - CurrentPos
  89 - SQL/MX DDL OP                    90 - GGSSQLCol
 100 - GGSPurgedata                    101 - GGSPurgeFile
 102 - GGSCreateFile                   103 - GGSAlterFile
 104 - GGSRenameFile                   105 - GGSSetmode
 107 - GGSControl                      106 - GGSChangeLabel
 160 - DDL OP                         115 - GGSKeyFieldComp
 117 - GGSKeyFieldComp32               161 - RecordFragment
 116 - LargeObject                   132 - GGSCreateSequence
 133 - GGSAlterSequence               134 - GGSDropSequence
 150 - RestartAbend                    151 - RestartOK
 152 - RecoveryEnd                     200 - GGSBulkio
 201 - GGSFileClose                    202 - GGSLoggerTS
 203 - GGSExtractTS                    204 - GGSCollectTS
 205 - GGSComment                      250 - LibOpenTrace
 251 - LibCloseTrace                  252 - LoggerOpenTrace
 253 - LoggerCloseTrace               254 - LoggerAddedInfo
 249 - LoggerAddedStats
```

SKIP

Use SKIP to skip the specified number of records.

Default None

Syntax `SKIP <n>`

Argument	Description
`<n>`	The number of records to skip.

Example `SKIP 50`

TIME

Use TIME to display the current time in local and GMT formats.

Default None

Syntax `TIME`

TIMEOFFSET

Use TIMEOFFSET to set the Logdump time format. Without arguments, TIMEOFFSET displays the current time offset. Options enable you to set the time to the local time, Greenwich Mean Time (GMT), or a specific offset from GMT. The specified time format applies to the timestamps shown in records as well as any Logdump commands that accept a time string argument.

Default `LOCAL`

Syntax `TIMEOFFSET {LOCAL | GMT | GMT + <hh[:mm]> | GMT - <hh[:mm]>}`

Argument	Description
`LOCAL`	Sets the time to that of the local system.
`GMT`	Sets the time to Greenwich Mean Time (GMT).
`GMT + <hh[:mm]>`	Sets the time ahead of GMT by the specified number of hours and, optionally, minutes.
`GMT - <hh[:mm]>`	Sets the time behind GMT by the specified number of hours and, optionally, minutes.

Example `TIMEOFFSET GMT -01`

TRAILFORMAT

Use TRAILFORMAT to set the version of the GoldenGate trail or extract file that is being saved when using the SAVE command.

Default `NEW`

Syntax `TRAILFORMAT {NEW | OLD}`

Argument	Description
`NEW`	Sets the format to that used by GoldenGate version 6.0 and later.
`OLD`	Sets the format to that used by GoldenGate versions earlier than 6.0.

TRANSBYTELIMIT

Use TRANSBYTELIMIT to prevent normal-sized transactions from being tracked in the transaction table specified with the TRANSHIST command. It sets a lower boundary for the number of bytes in a transaction and should be set to represent a normal-sized transaction for the environment being evaluated with Logdump. Setting a boundary reduces the amount of data that is stored and, consequently, the amount that must be reviewed when troubleshooting.

Default 10000 bytes

Syntax `TRANSBYTELIMIT <n>`

Argument	Description
<n>	The number of bytes in a normal-sized transaction.

Example `TRANSBYTELIMIT 9000`

TRANSHIST

Use TRANSHIST to keep track of the size of transactions in a trail or file. Logdump tracks the transactions in an internal history table in descending order according to the number of bytes of data in each one. When the history table is full, the smallest transaction is removed to allow a larger transaction to be added to the list.

Use TRANSHIST in conjunction with other Logdump commands to determine whether or not your applications generate large transactions and to identify their relative size. This information can be used when deciding how to group tables into different processing groups for faster throughput. For more information, see page 81.

NOTE You can use the SEND EXTRACT command with the SHOWTRANS option to view a list of long-running transactions. Other options enable you to control whether those transactions are ignored or processed by GoldenGate.

Default 0 (do not maintain history)

Syntax `TRANSHIST <n>`

Argument	Description
<n>	Sets the size of the history table, in bytes. Valid values are 0 through 200 bytes. A value of 0 means that no transaction history is maintained.

Example `TRANSHIST 150`

TRANSRECLIMIT

Use TRANSRECLIMIT to prevent normal-sized transactions from being tracked in the transaction table specified with the TRANSHIST command. It sets a lower boundary for the number of records in a transaction and should be set to represent a normal-sized transaction for the environment being evaluated with Logdump. Setting a boundary reduces the amount of data that is stored and, consequently, the amount that must be reviewed when troubleshooting.

Default 100 operations

Syntax `TRANSRECLIMIT <n>`

Argument	Description
`<n>`	The number of records in a normal-sized transaction.

Example `TRANSRECLIMIT 90`

USERTOKEN

Use USERTOKEN to control whether or not user token data is displayed with each record. A user token is data specified by a GoldenGate user that is stored in the record header and can be mapped to a target column or used for some other purpose during GoldenGate processing.

Without arguments, USERTOKEN displays the status of user token display (ON or OFF). With the ON option, the name of the token and its length are displayed. The DETAIL option shows the actual token data.

Figure 10 User tokens, with data

```
User tokens        7 bytes
5465 7374 0031 00                | Test.1
```

Default Display token name and length.

Syntax `USERTOKEN {ON | OFF | DETAIL}`

Argument	Description
`ON`	Enables the display of user tokens.
`OFF`	Disables the display of user tokens.
`DETAIL`	Displays the token data.

WRITELOG

Use WRITELOG to write text to the session log. Before using this command, start logging with the LOG command (see page 112).

Default None

Syntax `WRITELOG <text>`

Argument	Description
`<text>`	Any text string. Quotes are optional.

Example `WRITELOG “Customer name is ABC Company.”`

X

Use X to execute a program from within Logdump. When you exit the program, the Logdump prompt returns.

Default None

Syntax `X <program> [<string>]`

Argument	Description
`<command>`	The program to run.
`<string>`	A character string, such as input arguments.

Example The following series of commands and output shows how you can exit Logdump, issue other commands from the shell or within GGSCI, and then return to the Logdump command line.

```
Logdump 696 >x ggsci

GoldenGate Command Interpreter
Version .....

GGSCI (sysa) 1> status er *
GGSCI (sysa) 2> start er *
GGSCI (sysa) 3> info er *
GGSCI (sysa) 4> exit
Logdump 697 >
```

CHAPTER 5

Event Error and Warning Messages

This appendix describes messages for the Extract, Replicat, Manager and Collector processes.

ERR-STARTUP-PARAM — 101

Cause An unrecognized or missing parameter was encountered during program startup.

Recovery Correct the parameter file and restart.

ERR-FILE — 103

Cause A file error was encountered during an operation on the file or process.

Recovery Examine the error number to determine the exact nature of the error (for example, a security violation or disk is full). Attempt to correct the error.

ERR-MEM-ALLOC —105

Cause There was an internal problem allocating memory.

Recovery Restart the process. If the problem persists, contact GoldenGate Support.

ERR_BAD_SECURITY — 108

Cause An error occurred processing the CMDSEC file, or there was unauthorized access to a GoldenGate program.

Recovery Modify security settings in the CMDSEC file as necessary.

ERR-SQL— 109

Cause An error was encountered during a SQL operation.

Recovery Dependent on the type of error. If the error occurred during a replicated operation, error detail can be found in the Replicat discard file. Error detail indicating the exact nature of the database error can generally be found in the report file for the process as well (using the VIEW REPORT command).

ERR-COMPILE-WHERE— 111

Cause Incorrect syntax was specified for a WHERE, COLMAP, or KEYCOLS statement in the parameter file.

Recovery Correct the syntax and restart the process.

ERR-EXTRACT-FILE —112

Cause An error was encountered relating to processing a trail file. Further information is included about the cause in the error message.

Recovery Dependent on the error message. If this is a file error (such as a security violation), correct the problem and restart the process.

ERR_USER_EXIT — 117

Cause GoldenGate failed to load or execute a user exit.

Recovery Make certain that the correct shared library and function name are specified in the parameter file

ERR-DATABASE — 118

Cause An error was encountered during a database operation (possibly an ODBC operation or a native database error).

Recovery Dependent on the type of error. If the error occurred during a replicated operation, more error detail can be found in the Replicat discard file. Error detail indicating the exact nature of the database error can generally be found in the report file for the process as well (using the VIEW REPORT command).

ERR_DATA_TYPE — 120

Cause Unrecognized data type.

Recovery Make certain that tables being processed only contain supported data types.

ERR_ASCII_FORMATTING — 121

Cause An error occurred converting a record to ASCII format.

Recovery Examine the error detail to determine the exact nature of the problem. Use this information to modify the ASCII formatting options in the parameter file.

ERR-PROCESS-CREATE —130

Cause Manager encountered an error trying to create another process. The error message provides more detail as to the cause.

Recovery Fix the underlying cause of the problem. The cause may be a security violation, a missing program file or other reason.

ERR-TCPIP — 150

Cause A TCP/IP error occurred while attempting to write extracted data to a remote system.

Recovery If the error is "Connection Refused," make sure that the Manager process has been started on the remote system. Also, make sure the PORT argument of the RMTHOST parameter matches the remote Manager PORT parameter. TCP/IP errors usually are retried automatically for a period of time.

ERR-CHECKPOINT —151

Cause A file error was encountered during an operation on the checkpoint file.

Recovery Examine the error number to determine the exact nature of the error (for example, a security violation or disk is full). Attempt to correct the error and restart the process.

ERR_TRACE — 152

Cause An error occurred opening or writing to a trace file.

Recovery Evaluate the error detail to determine the exact nature of the problem. For example, there may be a permissions problem, or the disk may be full.

ERR-BAD-RECS —160

Cause An error was encountered replicating one or more records, and the record was discarded. By default, this message is generated only once per every 100 records discarded.

Recovery Examine the discard file to determine the exact cause of the error. The cause of this error is either a data mapping problem or a database error (such as a duplicate or missing record). The discard file also contains the contents of the record, which can be used to manually recover the data, if desired. Recovery is application-specific and action is not always required.

ERR-MAX-DISCARD-RECS —161

Cause The discard file has reached its maximum size and cannot hold any more discard records.

Recovery Several actions can be taken:

- Ignore the condition and allow processing to continue.
- Increase the MEGABYTES option of the DISCARDFILE parameter, and restart the process.

Specify DISCARDROLLOVER (which schedules the creation of new discard files on a periodic basis), and restart the process.

ERR-MAPPING-TO-TANDEM —162

Cause The Extract process was unable to map data from a row in the source database into the corresponding table or file format on the NonStop system.

Recovery The discard file contains the contents of the source record. Recovery procedures can include:

- Specifying compatible columns in the map and restarting replication.
- Manually fixing the data in the source database.
- Ignoring the problem.

ERR-NT-INTERNAL —170

Cause A Windows error was encountered by Manager.

Recovery Fix the error condition indicated by the reported message and restart Manager.

ERR_BAD_COMMAND —171

Cause An invalid command was received during inter-process communications.

Recovery Make certain that source and target versions of GoldenGate are the same.

ERR_INTERNAL_RESOURCE — 172

Cause An internal resource error occurred.

Recovery Contact GoldenGate Support.

ERR_EXCEEDED_BUFFER_SIZE — 173

Cause The size of a record exceeded the maximum internal buffer size.

Recovery Remove the table with the large record from replication or use a parameter to warn only.

ERR_THREAD_ERROR — 180

Cause An error occurred using threads.

Recovery Make certain that threads are supported by the platform.

ERR_TRANSACTION_ERROR — 181

Cause An error occurred processing a transaction.

Recovery Contact GoldenGate Support.

ERR-PROCESS-ABEND —190

Cause The Extract or Replicat process is terminating abnormally (abending).

Recovery Examine the previous error messages and the report file (using VIEW REPORT) to determine the exact cause.

ERR_LIBRARY_ABEND — 191

Cause An error occurred using the compression library.

Recovery Contact GoldenGate Support.

WRN-RECORD-VERSION — 208

Cause A record being processed by the Collector process does not match the definition of the record specified in the definitions file (the -d parameter to Collector).

Recovery If the definition of a source table or file has changed, run the DEFGEN program on the NonStop Server again, export the new definitions to the target platform, and restart Extract and/or Collector.

WRN_FILE_NO_AUDIT — 210

Cause Transactions for this table are not being captured in the transaction log.

Recovery Modify the table or database configuration to capture transactions for the table.

WRN-MISSING-INFO — 216

Cause Replicat could not evaluate a WHERE clause because one or more columns was missing from the source record.

Recovery Specify a WHERE clause that includes only columns that are always present in the record (such as primary key columns). Otherwise, you may need to specify uncompressed updates on the source platform.

WRN-FILE — 217

Cause A non-fatal file error was encountered during an operation on a file.

Recovery Examine the error number to determine the exact nature of the error (for example, a security violation or disk is full). Attempt to correct the error.

WRN-SQL —218

Cause A non-fatal SQL error occurred.

Recovery Dependent on the type of error. Evaluate the error detail. If the error occurred during a replicated operation, error detail can be found in the Replicat discard file. Error detail indicating the exact nature of the database error can generally be found in the report file for the process as well (using the VIEW REPORT command).

WRN-MISSING-TRANS-BEGIN — 221

Cause Replicat cannot process the current transaction because the beginning of the transaction is missing. Processing resumes at the next transaction. Possibly, a previous trail file was purged.

Recovery Examine the discard file for the rejected data and use manual procedures to recover the data.

WRN-MAPPING-PROBLEM — 222

Cause A problem was encountered while mapping a source record to a target format. Usually this means that the source record has unexpected or incorrect data in it. The record was discarded. Depending on how error handling is configured in the parameter file, the process continues, aborts the transaction, or ends abnormally.

Recovery Examine the discard record to determine what may have caused the mapping problem. Use manual procedures to extract and/or replicate the necessary data, or alter the mapping specification and restart.

WRN_TRANS_ABORT — 223

Cause A database recovery was detected and all in-progress transactions are aborted.

Recovery None required.

INF-ROLLOVER — 224

Cause A new trail file was opened by Extract.

Recovery None. Used to trace processing activity.

WRN-PURGE-EXTRACT — 226

Cause The Manager process purged a trail file that is no longer required by any replication processes.

Recovery None required. Informational.

WRN-DATABASE — 228

Cause An error was encountered during a database operation (possibly an ODBC operation or a native database error).

Recovery Dependent on the type of error. If the error occurred during a replicated operation, error detail can be found in the Replicat discard file. Error detail indicating the exact nature of the database error can generally be found in the report file for the process as well (using the VIEW REPORT command).

WRN-START-EXTRACT-FAILED — 230

Cause Manager failed to start Extract for the reason included as part of this message (usually a missing file or security violation).

Recovery Correct the condition and restart Extract.

WRN-START-REPLICAT-FAILED — 231

Cause Manager failed to start Replicat for the reasons included as part of this message (usually a missing file or security violation).

Recovery Correct the condition and restart Replicat.

WRN_MGR_START_BATCH_FAILED — 232

Cause The Manager process was unable to start a child process to perform background tasks.

Recovery Specify NOUSETHREADS in the Manager parameter file to force the parent process to perform background tasks.

WRN-STMT-MAX — 240

Cause Replicat for Oracle failed because an internal Oracle statement limit was reached.

Recovery Several actions can be taken:

- Consult the Oracle user documentation for instructions on how to increase the number of open cursors held by a program at any given time.
- Specify the MAXSQLSTATEMENTS parameter in the Replicat parameter file (down to a value of 20 or lower).
- Specify the NODYNSQL parameter (which allows replication to continue at a slower rate).

WRN-PROCESS-ABENDED — 250

Cause This message is reported periodically by Manager when an Extract or Replicat process has ended abnormally.

Recovery Review the cause of the abnormal termination in the process report file (using VIEW REPORT) and the error log. Correct the error condition and restart processes as appropriate.

WRN-FORCE-STOP — 251

Cause This message is generated by Extract and Replicat whenever the STOP command is issued with the ! (exclamation point) option, which causes Extract and Replicat to abort the current transaction, if one exists.

Recovery None. Used to trace processing activity.

WRN_FORCE_STOP — 251

Cause The program was forced to stop by a user.

Recovery No action required.

WRN-MGR-LAG-THRESHOLD — 260

Cause This message is generated by Manager whenever lag for the specified process exceeds a configured threshold. It is a warning that a problem may exist in the system.

Recovery Investigate the cause of possible performance problems (such as a process running at low priority or a faulty network).

WRN-COLL-OPEN-FILE-ERR —261

Cause A file error was encountered when Collector opened a file.

Recovery Examine the error number to determine the exact nature of the error (for example, a security violation or disk is full). Attempt to correct the error and restart Extract on the source system.

WRN_MGR_LAG_THRESHOLD — 260

Cause The process lag time has exceeded the warning threshold.

Recovery Evaluate the process in question to determine the cause of the lag.

WRN-SERVICE-RQST-STOP — 270

Cause Manager has received a request from the Windows Control Services Manager to stop.

Recovery None. Used to trace processing activity.

WRN_SERVICE_RQST_PAUSE — 271

Cause The Windows Control Services Manager requested a pause.

Recovery No action required.

INF_SERVICE_RQST_CONT — 272

Cause The Windows Control Services Manager requested a continue.

Recovery No action required.

WRN-SERVICE-RQST-SHUTDOWN —273

Cause Manager has received a request from the Windows Control Services Manager to stop because of a general system shutdown.

Recovery None. Used to trace processing activity.

WRN_WIN32_API_CALL_ERR — 274

Cause Windows failed to return information about child processes.

Recovery No action required.

WRN_COLUMN_VALUE_TRUNCATED — 275

Cause A column value was truncated because of a limitation on the maximum column length.

Recovery Increase the maximum column length by using the MAXLONGLEN parameter.

WRN_MEM_THRESHOLD — 276

Cause GoldenGate is approaching memory threshold for processing stored procedures.

Recovery Reduce the number of parameters passed by the stored procedure.

WRN_FILTER_NOT_PASSED — 280

Cause The FILTER clause did not pass, so the record was ignored.

Recovery Modify the FILTER clause.

INF-PROCESSED-MARKER — 300

Cause An audit marker record created by GGSCI was processed by Replicat.

Recovery None. Markers are used to mark application-specific points of interest within the audit trails on NonStop Server.

INF_MGR_STARTING_PROCESS — 302

Cause The Windows Control Services Manager requested a start.

Recovery No action required.

INF-USER-CMD — 301

Cause A command was received by the Extract or Replicat process from GGSCI.

Recovery None. Used to trace processing activity.

INF-TCP-ERR-RECOVERED — 305

Cause A TCP/IP operation that previously encountered an error was retried successfully.

Recovery None. Indicates that processing has resumed normally after problem correction.

INF-EXTRACT-STARTED — 310

Cause Extract started.

Recovery None. Used to trace processing activity.

INF-EXTRACT-RESTARTED — 311

Cause Extract was restarted by Manager after ending abnormally.

Recovery None. Used to trace processing activity.

INF-EXTRACT-STOPPED — 312

Cause Extract stopped normally (at the request of GGSCI or by parameter specification).

Recovery None. Used to trace processing activity.

INF-REPLICAT-STARTED — 320

Cause Replicat started.

Recovery None. Used to trace processing activity.

INF-REPLICAT-RESTARTED — 321

Cause Replicat was restarted by Manager after ending abnormally.

Recovery None. Used to trace processing activity.

INF-REPLICAT-STOPPED — 322

Cause Replicat stopped normally (at the request of GGSCI or by parameter specification).

Recovery None. Used to trace processing activity.

INF-MGR-STARTED — 330

Cause Manager started.

Recovery None. Used to trace processing activity.

INF-MGR-STOPPED — 331

Cause Manager stopped normally (at the request of GGSCI or by parameter specification).

Recovery None. Used to trace processing activity.

INF-PURGE-OLD-ROWS — 340

Cause Manager deleted unneeded rows from the GoldenGate log table (as the result of the PURGEOLDEXTRACTS parameter).

Recovery None. Informational.

INF-PROCESS-RUNNING — 350

Cause Manager is reporting that a process is alive.

Recovery None required. Used to trace processing activity.

INF-MGR-PROCESS-DOWN-OK — 351

Cause Manager is reporting that a process is down and terminated normally.

Recovery None required. Used to trace processing activity. If this message is marked critical, restarting the process may be required.

INF-MGR-LAG-THRESHOLD — 360

Cause This message is generated by Manager whenever lag for the specified process exceeds a configured threshold.

Recovery This message is most often generated for status only and usually does not require recovery.

INF-MGR-AUTOSTART — 361

Cause This message is generated by Manager at startup if the AUTOSTART parameter was configured. (AUTOSTART automatically starts specified GoldenGate processes.)

Recovery None. Informational.

INF-MGR-BOOTDELAY — 362

Cause This message is generated by Manager at startup if the BOOTDELAYMINUTES parameter was configured. (BOOTDELAYMINUTES delays further Manager activities on Windows until the system has been running for a specified number of minutes.)

Recovery None. Informational.

INF_RETRY_REP_OP — 363

Cause GoldenGate is retrying a SQL operation.

Recovery No action required.

INF_FILE_ERR_RECOVERED — 364

Cause GoldenGate recovered from a retried SQL operation.

Recovery No action required.

GGS_EVT_COLL_OPENING_FILE — 370

Cause Collector opened a batch file on an IBM z/OS system.

Recovery No action required.

GGS_EVT_COLL_CLOSING_FILE — 371

Cause Collector closed a batch file on an IBM z/OS system.

Recovery No action required.

GGS_EVT_COLL_TERMINATING — 372

Cause Collector is shutting down after client disconnect.

Recovery No action required.

GGS_EVT_COLL_LISTENING — 373

Cause Collector has started and is listening for requests.

Recovery No action required.

INF_CTRL_EVENT_RECEIVED — 374

Cause GoldenGate received a CLOSE, SHUTDOWN, or LOGOFF event from Windows.

Recovery No action required.

INF_TRACE_STARTED — 375

Cause GoldenGate opened a trace file.

Recovery No action required.

INF_TRACE_STOPPED — 376

Cause GoldenGate closed a trace file.

Recovery No action required.

GGS_EVT_COLL_TRUNCATED_FILE — 377

Cause Collector truncated a batch file on an IBM z/OS system.

Recovery No action required.

GGS_EVT_COLL_CLOSE_ACTION — 378

Cause The predefined close action has been executed for a batch file on the IBM z/OS system.

Recovery No action required.

INF_GGSCI_CMD — 399

Cause A command was issued through GGSCI.

Recovery No action required.

EXTRACT READ, ERROR 13 (PERMISSION DENIED) — 501

Cause Extract was unable to open the redo log file. Indicates that Extract does not have read permission on the file.

Recovery Grant operating system permission to the user running Extract to read the redo log files. This user also must have read permission to the archive files.

EXTRACT READ, KEY COLUMN(S) MISSING FROM UPDATE FOR ... — 509

Cause Key-column values for updates are not available.

Recovery Enable replication or supplemental logging (depending on the database). See *GoldenGate for Windows and UNIX Administrator Guide* for database-specific requirements for making key values available to GoldenGate.

ERR_THREAD — 990

Cause An error occurred using threads.

Recovery Make sure threads are supported by the platform.

Glossary

The following explains terminology contained in this manual.

Term	Definition
abend	*Abnormal end*. The failure or unexpected termination of a process running on a computer system.
after image	The values of a row in a database after an insert or update is performed.
alias Extract	An Extract group that operates on a target system that resides within a more secure network zone than the source system. The purpose of the alias Extract is to initiate TCP/IP connections from the target to the less-trusted source. Once a connection is established, data is processed and transferred across the network in the usual manner by a passive Extract group that operates on the source system.
append mode	The default method of writing to the trail, whereby Extract appends reread data to the trail file after a failure, instead of overwriting the old data.
Archived Log Only mode (ALO)	A mode of operation for Extract, where the process is configured to read exclusively from the archived transaction logs on a production or standby database system.
audit trail	A file on a NonStop Server system that stores modifications made to a database for the purpose of replication and recovery.
batch run	A one-time processing run that has a distinct beginning and an end, as opposed to continuous processing that does not have a specific end point, such as online change synchronization.
before image	The values that exist in a row in a database before a SQL operation is performed on that row.
bidirectional synchronization	Permits load distribution across multiple databases and servers where, in most cases, different users can change the same sets of data and those changes are synchronized by GoldenGate.
BLOB	See *LOB*.
caller	The GoldenGate process that executes a user exit routine.

Term	Definition
cascading synchronization	A GoldenGate configuration in which data is sent from a source system to one or more intermediary systems and, from those systems, to one or more other systems in a synchronized state.
change synchronization	The process of synchronizing data changes made to a database on one system with a similar set of data on one or more other systems.
checkpoints	Internal indicators that record the current read and write position of a GoldenGate process. Checkpoints are used by the Extract and Replicat processes for online change synchronization to ensure data accuracy and fault tolerance.
checkpoint file	A file on disk that stores the checkpoint generated by GoldenGate processes.
checkpoint table	A table created in the target database that maintains Replicat checkpoints, used optionally in conjunction with a standard checkpoint file on disk.
CLOB	See *LOB*.
CMDSEC file	A GoldenGate file that stores rules for GGSCI command permissions.
Collector	The process that receives data from the Extract process over TCP/IP and writes it to a trail or extract file on the target system.
collisions	Errors that occur when data changes that are replicated by GoldenGate are applied to a target table, but the target row is either missing or is a duplicate.
column	One among a set of attributes assigned to an entity that is described by a database table. For example, there can be columns for the name, address, and phone number of the entity called “employees.”
column-conversion functions	Built-in GoldenGate processing functions that perform comparisons, tests, calculations, and other processing for the purpose of selecting and manipulating data.
column map	See *map*.
compressed update	A method of logging SQL update operations by which only column values that changed as the result of the update are logged to the transaction log.
conflict resolution	Instructions used in bidirectional synchronization that provide processing and error-handling rules in the event that the same SQL operation is applied to the same row in two or more databases at (or about) the same time.

Term	Definition
consolidated synchronization	The process of replicating different data from two or more databases to one central database, such as in data warehousing.
conversion	See *transformation*.
data definitions file	See *source definitions file* and *target definitions file*.
data pump	A secondary Extract process that reads from an extract file or trail. The trail is populated by a primary Extract process that reads from the data source.
data source	The container of the data changes that are to be processed by GoldenGate. A data source can be: ◆ the transaction log of a database ◆ a Vendor Access Module
data source name (DSN)	A DSN defines an ODBC connection to a database. A DSN consists of a database name, the database directory, the database ODBC driver name, database authentication information, and other information depending on the database. External applications, such as GoldenGate, require a DSN, because a DSN enables an application to connect to a database without having to encode the required information within the application program. The three types of DSN are ◆ A system DSN can be used by any entity that has access to the machine. It is stored within the system configuration. ◆ A user DSN can only be used by a specific user. It is stored within the system configuration. ◆ A file DSN is stored in a text file with a .dsn extension. It can be shared among different systems where the required ODBC driver is installed.
data type	An attribute of a piece of data that identifies what kind of data it is and what kinds of operations can be performed on it. For example, an integer data type is a number, and a character data type contains letters.
DDL	*Data Definition Language*. Data that defines the structure of a database, including rows, columns, tables, indexes, and database specifics such as file locations, users, privileges, and storage parameters.
DDLGEN	A GoldenGate utility that generates table-creation instructions based on the source tables. These instructions are used to create target tables.
DEFGEN	A GoldenGate utility that generates a data definitions file.

Term	Definition
Director	Also known as GoldenGate Director, a graphical user interface that enables users to monitor and manage GoldenGate processes. The components of Director are: **Director Administrator**: A utility used by administrators to define users and instances of GoldenGate. **Director Server**: A software module that gathers data about the GoldenGate processes. **Director Client**: Software installed on a user’s system as an interface to Director. **Director Web**: A browser-based user interface to Director (requires no software to be installed).
discard file	A GoldenGate file containing information about SQL operations that failed. This file is created when a record cannot be processed, but only if the DISCARDFILE parameter exists in the parameter file to specify the location for the file.
DML	*Data Manipulation Language*. Retrieves and manipulates data in a database. In the case of SQL, the actions are “select”, “insert”, “update”, and “delete”.
DSN	See data source name (DSN).
dynamic Collector	A Collector process that the Manager process starts automatically, as opposed to a static Collector.
EMSCLNT	A GoldenGate utility that distributes GoldenGate system error messages that originate on Windows and other supported operating systems to the EMS (Event Management Subsystem) server on the NonStop Server.
ENCKEYS file	A GoldenGate lookup file that stores encryption keys.
encryption	A method of encoding data into a format that is unreadable to anyone except those who posses a password or decryption code to decipher it.
error log	A file that shows processing events, messages, errors, and warnings generated by GoldenGate. Its name is ggserr.log and it is located in the root GoldenGate directory.
event marker system	A system that customizes GoldenGate to take a specific action during processing based on a record that qualifies for filtering criteria. For example, you can skip the record or stop the GoldenGate process when the record is encountered. See also event record.

Term	Definition
event record	A record in the transaction log that satisfies specific filter criteria and is used to trigger a specific action during processing. See also event marker system.
exceptions map	A special MAP parameter used specifically for error handling, which executes only after an error and sends error data to an exceptions table.
exceptions table	A database table to which information about failed SQL operations is written as the result of an exceptions map. Used for error handling.
Extract	The GoldenGate program that reads data either from a data source, from source tables, or from a local trail or file. Extract processes the data for delivery to the target system. A *primary Extract* reads the data source or database tables, and a *data-pump Extract* reads a local trail that is populated by a primary Extract.
extract file	A file written by GoldenGate where data is stored temporarily awaiting further processing during a batch run or initial load.
extraction	The processing of reading data from database tables or from a data source in preparation for further processing and/or transmission to a target database.
fetch	A query to the database issued by the Extract process when processing a record from the transaction log. A fetch is required if the data values that are needed to complete the SQL operation are not present in the record.
file header	See header.
filtering	The use of rules to select and exclude data for extraction or replication.
function	A segment of code that can be executed within an application or routine. See also *column-conversion functions*.
GGSCI	*GoldenGate Software Command Interface*. The primary interface for issuing commands that configure, control, and monitor GoldenGate.
GLOBALS file	A text file in the root GoldenGate directory that contains parameters which apply to the GoldenGate instance as a whole, as opposed to runtime parameters that are specific to a process such as Extract or Replicat.
GoldenGate Rollback	A utility that uses before images to undo changes made to a database.
group	Also known as *process group*. A group consists of a GoldenGate process (either Extract or Replicat) and the parameter file, the checkpoint file, and any other files associated with that process.

Term	Definition
header	A header can be: ◆ **A record header**: an area at the beginning of a record in a GoldenGate trail file that contains information about the transaction environment for that record. ◆ **A file header**: an area at the beginning of each file in a trail, or at the beginning of an extract file. This header contains information about the file itself, such as the GoldenGate version.
heterogeneous	A data environment where data is being exchanged among different types of applications, different types of databases, or different operating systems, or among a combination of those things.
homogeneous	A data environment where data is being exchanged among identical types of applications, databases, and operating systems.
initial load	The duplication of source data into a target database to make the two databases identical.
intermediary system	A system on the network that serves as a transfer station between the source and target systems. This system can be host to additional processing activities, such as transformation.
key	A column or columns in a table that are being used as a unique identifier for the rows in that table. GoldenGate uses the key to find the correct row in the target database and for fetches from the source database. For GoldenGate, a key can be the primary key, a unique key, a substitute key, or all of the columns of a table in the absence of a defined identifier.
KEYCOLS	A clause in a TABLE or MAP statement that defines a column or columns for GoldenGate to use as a unique identifier to locate any given row in a table.
KEYGEN	A GoldenGate utility that generates encryption keys.
lag	Extract lag is the difference between the time that a record was processed by Extract and the timestamp of that record in the data source. Replicat lag is the difference between the time that the last record in a trail was processed by Replicat and the timestamp of the record in the trail.
latency	The difference in time between when a change is made to source data and when that change is reflected in the target data.

Term	Definition
LOB	*Large Object*. A data type in a database that represents an unstructured object that is too large to fit into a character field, such as a Microsoft Word document or a video or sound file. Subsets of LOB are CLOB (Character Large Object) and BLOB (Binary Large Object), which contain character data and binary data, respectively.
log-based extraction	A method of extracting data changes from the database transaction log.
logical name	A name for a stored procedure that represents an instance of the *execution* of the procedure, as opposed to its actual name. For example, logical names for a procedure named "lookup" might be "lookup1," "lookup2," and so forth.
LUW	*Linux, UNIX, Windows*. An acronym that describes an application that runs on any of these platforms, such as DB2 LUW.
macro	A computer program that automates a task, such as the implementation of parameters and commands.
map	An association between a set of source data and a set of target data. A map can include data selection and conversion criteria. These maps are specified in a Replicat MAP parameter.
MAP statement	A Replicat parameter that specifies the relationship between a source table and a target table and the processing rules for those tables.
Manager	The control program for GoldenGate processing.
marker	A record that is inserted into the audit trail on a NonStop Server to identify application-specific events in the context of Extract and Replicat processing. See also event marker system.
object	For the purpose of this documentation, the term *object* refers to any logical component of a database that is visible to, and can be created by, its users for the purpose of storing data (for example, tables), defining ownership and permissions (for example, roles), executing an action on another object (for example, triggers), and so forth.
object record	A file containing attributes of the tables and other database objects that are configured for processing by GoldenGate, such as column IDs and data types.
ODBC	*Open Database Connectivity*. Acronym for a standard interface that enables applications to connect to different types of databases in a uniform manner. The goal of ODBC is to make the process of connecting to a database independent of programming languages, database systems, and operating systems.
online Extract	An Extract group that is configured for online change synchronization.

Term	Definition
online change synchronization	A GoldenGate processing method in which Extract and Replicat processes run continuously to synchronize data changes unless they are stopped by a GoldenGate user. Online processes maintain checkpoints in the trail.
online processing	See *online change synchronization*.
online Replicat	A Replicat group that is configured for online change synchronization.
operation	A single unit of work. This typically refers to a SQL change made to data or a change made to the structure of an object in the database, but can also refer to any work done by a computer process.
owner	A logical namespace in a database to which database objects are assigned as part of the organizational hierarchy. Because the ownership of database objects is managed differently by different database types, the term *owner* is used in this documentation to denote whichever entity is recognized by the database as the qualifier of an object name, typically a user or schema name. For example, in a qualified Oracle table name of scott.emp, the owner is scott.
overwrite mode	A method of writing data to the trail that was used in GoldenGate versions prior to version 10.0. In this mode, Extract overwrites existing data upon recovery, instead of appending it to the end of the trail file.
parameter	An input or output value for a computer program, such as the code of an application like GoldenGate, a stored procedure, a macro, script, or other processing instructions.
parameter file	A file containing parameters that control the behavior of a GoldenGate process. The default location for parameter files is the dirprm directory in the GoldenGate installation directory.
passive Extract	An Extract process that operates on the source system when an alias Extract is being used on the target. This GoldenGate configuration is required when security rules do not permit TCP/IP connections to be initiated from the source system (as a typical Extract would do) because the target is inside a more secure network zone. The passive Extract is the data pump, when one is being used; otherwise, it is the primary Extract.
pass-through data pump	A data pump that is configured with the PASSTHRU parameter to bypass the need to look up data definitions. This enables faster processing and enables a pump to be used on an intermediary system that has no database.
pass-through Extract	See *pass-through data pump*.

Term	Definition
primary Extract	An Extract group that reads from the data source or directly from the database tables. A primary Extract can write to a local trail, which is then read by a data pump Extract, or it can send the data across TCP/IP to the target system.
primary key	An integrity constraint consisting of a column or columns that uniquely identify all possible rows that exist in a table, current and future. There can be only one primary key for a table. A primary key contains an implicit NOT NULL constraint.
process report	A report generated for Extract, Replicat, and Manager that provides information about the process configuration and runtime statistics and events. The default location for process reports is the dirrpt directory of the GoldenGate installation directory.
record	A unit of information in a transaction log or trail that contains information about a single SQL operation performed on a row in a database. The term *record* is also used to describe the information contained in a specific row of a table.
record header	See header.
remote file	An extract file on a remote system.
remote trail	A trail on a remote system.
Replicat	The GoldenGate process that applies data to target tables or moves it to another application or destination.
report	See *process report*.
replication	The process of recreating source database operations and applying them to a target database.
report file	See *process report*.
rollover	The closing of one file in a sequence of files, such as a trail, and the opening of a new file in the sequence.
routine	A segment of code that is executed within an application such as GoldenGate, which calls functions that retrieve and return values and provide responses. See also *user exit*.
row	Information about a single instance of an entity, such as an employee, that is stored within a database table. For example, a row stores information about “John Doe” in relation to the broader collection of rows that stores information about John and the other employees in a company. Also commonly known as a *record*.

Term	Definition
source	The location of the original data that GoldenGate will be extracting, as in *source database* and *source system*.
source definitions file	A file containing the definitions of the source tables, which is transferred to the target system. This file is used by the Replicat process for data conversion when the source and target tables are dissimilar.
special run	See *batch run*.
statement	An elementary instruction in a computer programming language, for example a SQL statement, parameter statement, or command statement.
static Collector	A Collector process that is started manually by a GoldenGate user, instead of being started automatically by the Manager process.
stored procedure	A group of SQL, PL/SQL, or Java statements that are stored in the database and called on demand by a process or application to enforce business rules, supplement application logic, or perform other work as needed.
substitute key	A unique identifier that consists of any columns in a table that can uniquely identify the rows in that table. A substitute key is not defined in the definition of a table; it is created by creating a KEYCOLS clause in a TABLE or MAP statement.
synchronization	The process of making or keeping two or more sets of data consistent with one another. To be consistent, one set might be identical to the other, or one set might be a reorganized, reformatted, or expanded version of the other, while retaining the essence of the information itself.
table	A logical unit of storage in a database that consists of rows and columns, which together identify the instances of a particular entity (for example, “employees”) and the attributes of that entity, such as name, address, and so forth.
TABLE statement	An Extract parameter that specifies a source table or tables whose data is to be extracted from the database.
TAM (Teradata Access Module)	An interface between the Change Data Capture (CDC) component of a Teradata database and the Extract process. It allows GoldenGate to communicate with the Teradata replication components.
target	The destination for the data that is processed by GoldenGate, as in *target database* and *target system*.

Term	Definition
target definitions file	A file containing the definitions of the target tables. This file is transferred to the source system and is used by the Extract process for data conversion when the source and target tables are dissimilar.
task	A special type of batch run in which the Extract process communicates directly with the Replicat process over TCP/IP instead of using a Collector process or trail.
token	A user-defined piece of information that is stored in the header portion of a record in the GoldenGate trail file. Token data can be used to customize the way that GoldenGate delivers information.
trace table	A special table created for use by GoldenGate in an Oracle database. The table is used in conjunction with parameter settings to prevent replicated data from being sent back to the source in a bidirectional synchronization configuration.
trail	A series of files on disk where GoldenGate stores data temporarily in preparation for further processing. GoldenGate records checkpoints in the trail for online change synchronization.
transaction	A group of one or more SQL operations executed within a begin and end statement by a single user on a database.
transaction log	A set of files that records all of the SQL change operations performed on a database for the purpose of data recovery or replication.
transformation	Also called *conversion*. The process of manipulating source data to the format required by target tables or applications, for example converting dates or performing arithmetic calculations. You can do transformation by means of the GoldenGate column-conversion functions.
unidirectional synchronization	A configuration where data changes are replicated in one direction, source-to-target. Changes cannot be made to that same data and then sent back to the source, as is the case in a bidirectional configuration.
unique key	An integrity constraint consisting of a column or columns that uniquely identify all possible rows that exist in a table, current and future. Differs from a primary key in that it does not have an implicit NOT NULL constraint. There can be more than one unique key on a table.
universal data format	A proprietary data format that GoldenGate uses to store data in a trail or extract file. Universal data format allows data to be exchanged rapidly and accurately among heterogeneous databases.
user exit	A user-created program written in C programming code that is called during GoldenGate processing to perform custom processing such as to convert data, to respond to database events, and to repair invalid data.

Term	Definition
VAM (Vendor Access Module)	An API interface that is used by a GoldenGate process module, such as Extract or Replicat, to communicate with certain kinds of databases.
VAM trail	A series of files, similar to a transaction log, that are created automatically and aged as needed. Data operations from concurrent transactions are recorded in time sequence, as they occur, but not necessarily in transaction order. Used to support the Teradata maximum protection commit protocol.
wildcard	A placeholder for an unknown or unspecified character or set of characters. A wildcard is a means of specifying multiple names in a parameter or command statement. GoldenGate supports the asterisk (*) wildcard, which represents any number of unknown characters.

Index